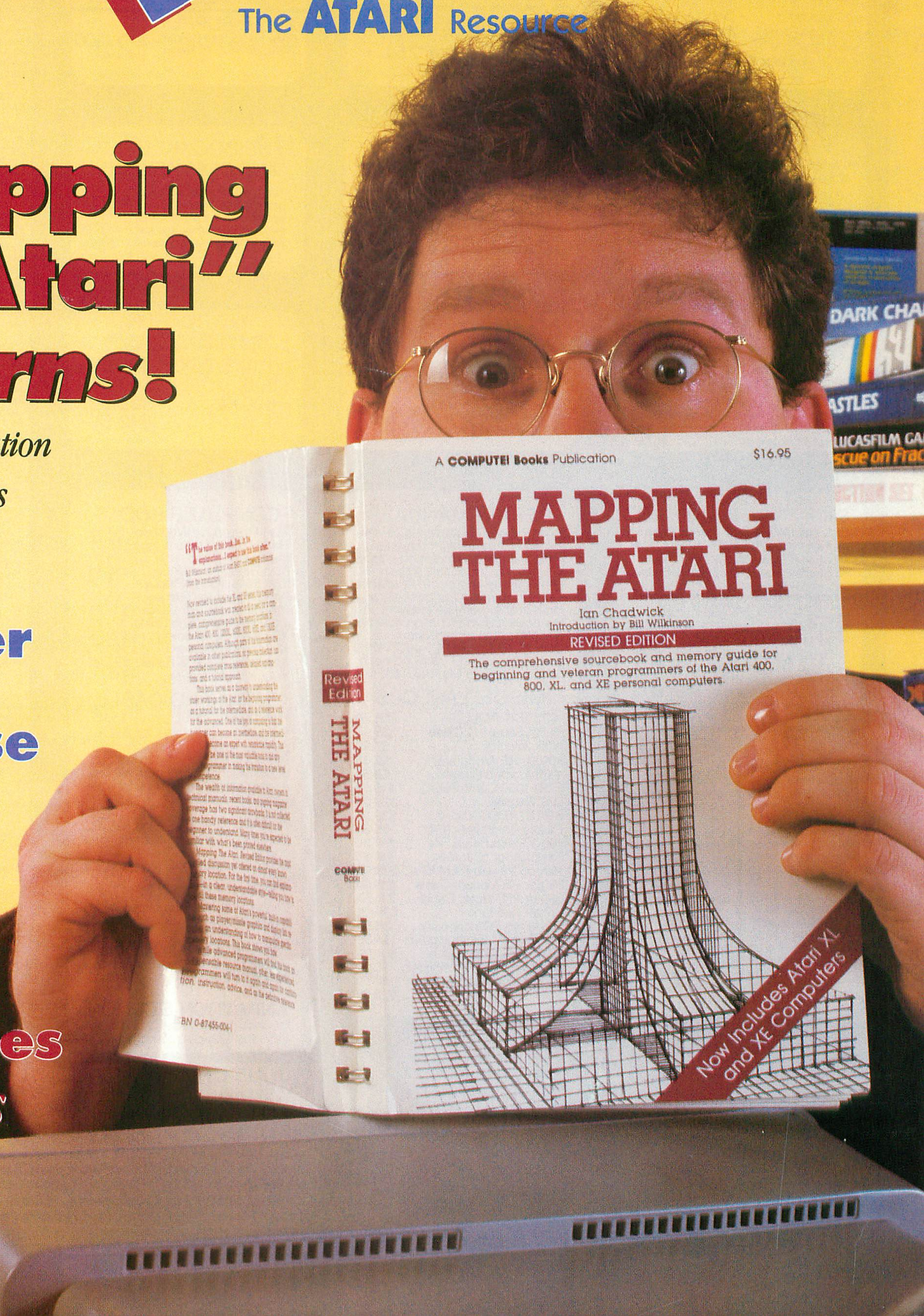
U.S.A. $3.95 CANADA $4.95
AUGUST 1989
VOLUME 8, NUMBER 4

Easy-To-Type Programs for XE XL 800

# Antic®

The ATARI Resource




## "Mapping the Atari" Returns!

*Exclusive serialization of Ian Chadwick's reference classic*

## Customer List Database

## New Atari Game Cartridges

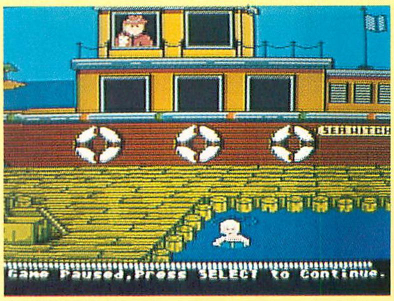
*11 Reviews*

0 74470 12728 08

Exclusive! Reference Classic Returns. page 22

# Antic®

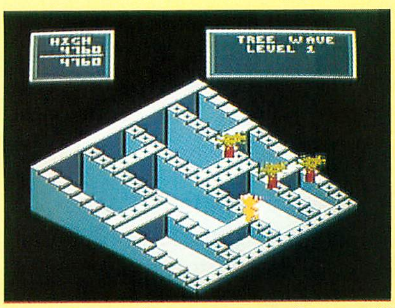
The ATARI® Resource










Carnival of Cartridges
Page 28

## FEATURES

AUGUST 1989, VOL. 8, NO. 4



## DEPARTMENTS



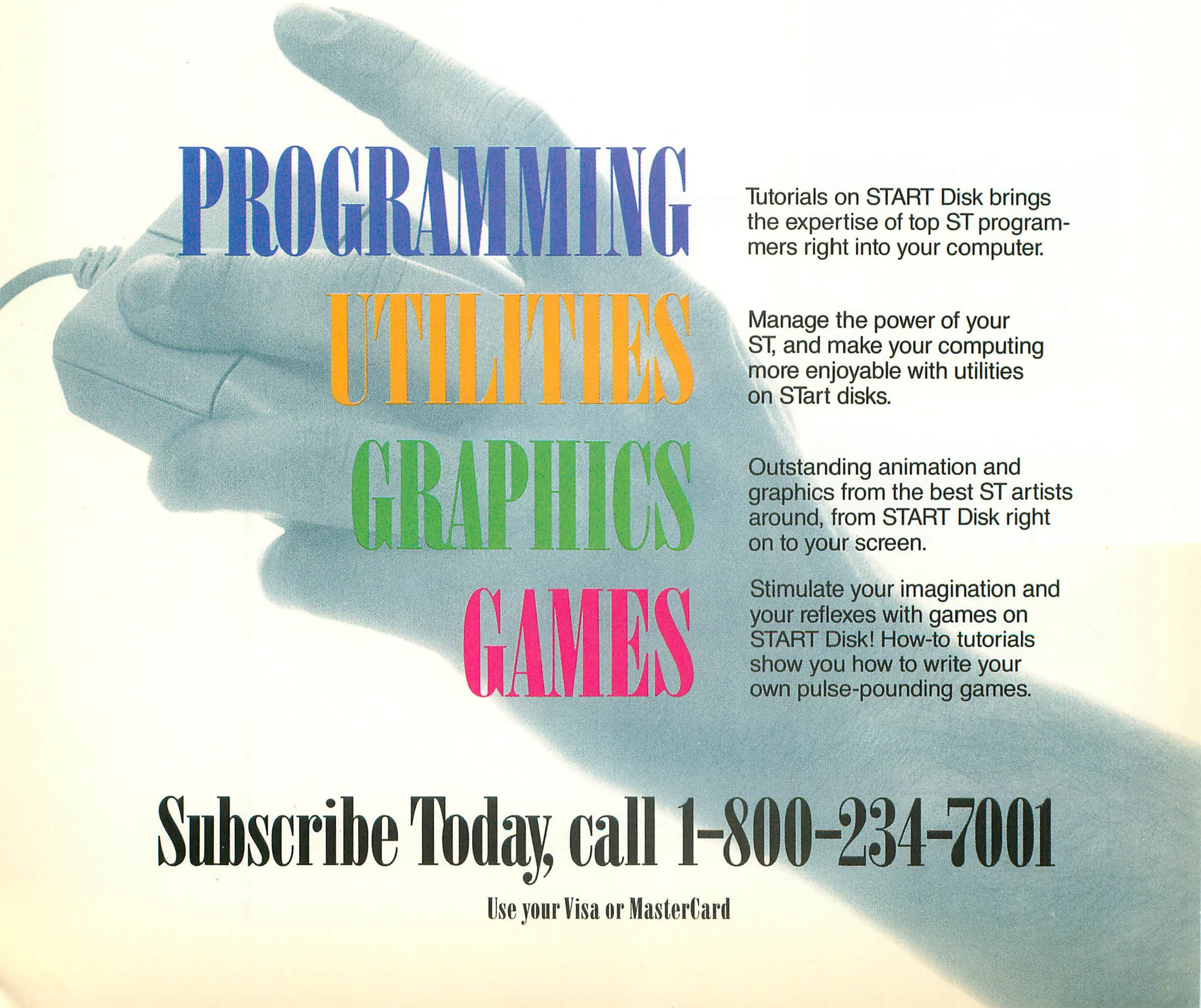
## SOFTWARE LIBRARY 5 Easy-To-Type 8-Bit Listings



**Antic**—The Atari Resource (ISSN 0745-2527) is published monthly by Antic Publishing. Editorial offices are located at 544 Second Street, San Francisco, CA 94107. ISSN 0745-2527. Second Class Postage paid at San Francisco, California and additional mailing offices. POSTMASTER: Send address change to **Antic**, The Atari Resource, P.O. Box 1569, Martinez, CA 94553. Subscriptions: One year (12 issues) $28. Canada and Mexico add $8, other foreign add $12. Disk Edition (12 issues with disks) $59.95, all foreign add $25. (California residents add 6½% sales tax for disk subscriptions. Editorial submissions should include text and program listings on disk and paper. Submissions will be returned if stamped, self-addressed mailer is supplied. **Antic** assumes no responsibility for unsolicited editorial material. 

# Get Clicking With Your Atari ST!

## Subscribe to START magazine with disk



**PROGRAMMING**

Tutorials on START Disk brings the expertise of top ST programmers right into your computer.

**UTILITIES**

Manage the power of your ST, and make your computing more enjoyable with utilities on STart disks.

**GRAPHICS**

Outstanding animation and graphics from the best ST artists around, from START Disk right on to your screen.

**GAMES**

Stimulate your imagination and your reflexes with games on START Disk! How-to tutorials show you how to write your own pulse-pounding games.

## Subscribe Today, call 1-800-234-7001

Use your Visa or MasterCard

**Antic**

James Capparell
*Publisher*
John Taggart
*Associate Publisher*

**EDITORIAL**
Nat Friedland
*Editor*
Carolyn Cushman
*Assistant Editor*
Marta Deike
*Editorial Coordinator*

**ART**
Kathleen McKeown
*Creative Services Director*
Jim Warner
*Art Director*
Dwight Been
*Associate Art Director*
Marianne Dresser
*Design Production Assistant*
Georgia Solkov
*Photo Editor and Cover Photography*
Julianne Ososke
*Production Manager*
Kate Murphy
*Advertising Production Coordinator*

**CIRCULATION**
Manny Sawit
*Director*
Amber Lewis
*Subscription Coordinator*
Ann Dunlap
*Retail Sales*
Dennis Swan
*Distribution Coordinator*
(415) 957-0886

**ADVERTISING**
*Marketing Manager*
Diane Van Arsdall
*Eastern Sales Representatives*
Denny Riley
*Western Sales Representative*
Austin Holian
(415) 957-0886

**ANTIC PUBLISHING, INC.**
James Capparell
*President and Chairman of the Board*
Donald F. Richard
Richard D. Capparella
*Directors*
Lee Isgur
*Advisor to the Board*
John Taggart
*Vice President*
John Cady
*Controller*

**GENERAL OFFICES**
**(415) 957-0886**
**544 Second Street, San Francisco, CA 94107**

Credit Card Subscription and Catalog Orders
**(800) 234-7001, Visa or MasterCard Only**

**SUBSCRIPTION CUSTOMER SERVICE**
(415) 372-6002
Antic, P.O. Box 1569, Martinez, CA 94553

## BERSERK

I'm trying to use BSRk from the August 1988 Antic Disk, and it seems to have wrong and missing commands. The first problem was that the program would load only partially into the BSR Controller, then stop and lock up. Then the asterisks were not showing the correct days of the week in use. I seem to have corrected this problem by changing the (X–1) to (X–1)–.1 in line 6610.

Also, the program isn't sending the code to turn the appliance modules on and off, although the program seems to turn the lamps on and off all right.

Ed Sacks
Garden Grove, CA

*As it says in the BSRk article,* don't *run the program from the monthly disk. Copy it to another disk, then follow the rest of the instructions on page 21 of that issue. Also, this program was tested with* standard Atari BASIC, *so don't use Turbo BASIC or BASIC XL/XE.*—ANTIC ED

## ANTIC REFRESHES

I want to activate the unused 4K block of memory on my Atari 800 (for my own programs, only, of course), but I don't know if the ANTIC chip refreshes over 48K. Can you help me?

Ralph Munoz
Los Angeles, CA

*There is no memory in that 4K block you can use—unless you have a memory upgrade that uses that area, in which case it will be refreshed.* — ANTIC ED.

## LAPTOP HOOKUP

How can I connect my Atari 800XL to my Tandy Model 200?

William Higginson, Jr.
Mayfield, KY

*See* Laptop-To-Atari Connection *in the November 1987* ***Antic.***—ANTIC ED

## OVER-PROTECTED INDUS

I have an Indus GT disk drive for my 130XE. The drive seems to be stuck in protect mode. Unless I can find a way to bypass or disable the protect mode, I am unable to save anything to disk. A letter to Indus Systems was returned unopened. Can you help me?

Vinton Henderson
APO NY

*Future Systems Inc. does repair work on Indus drives. Their address is 21634 Lassen Street, Chatsworth, CA 91311. (818) 407-1647.*—ANTIC ED

## CHALKBOARD POWERPAD

In your October I/O board you had a reader who wanted to know where he could get the ChalkBoard PowerPad. We have several dozen units in stock priced at $30 each. We also have Music Maestro and Bear Jam PowerPad cartridge/overlay software available at $15 each. (The PowerPad is useless alone and requires a plastic overlay plus software.)

If any of your I/O Board readers could find the Micro Illustrator cartridge and overlay (ChalkBoard PowerPad version only!) they would have an excellent drawing pad due to the large drawing surface of the PowerPad.

Gail Maddox
American Techna-Vision
15338 Inverness Street
San Leandro, CA 94579
(415) 352-3787

***Antic** welcomes your feedback, but we regret that the large volume of mail makes it impossible for the Editors to reply to everyone. Although we do respond to as much reader correspondence as time permits, our highest priority must be to publish I/O answers to questions that are meaningful to a substantial number of readers.*

***Send letters to: Antic I/O Board, 544 Second Street, San Francisco, CA 94107.***

NEW ARRIVALS

# NEW CARTRIDGES!

STAR RAIDERS
E.T.
DIG DUG
ASTEROIDS
DEFENDER
MISSLE COMMAND
GALAXIAN
FACEMAKER

## 1020 COLOR PRINTER PLOTTER

Complete with:
- 2 Pen Sets
- 1 Roll Paper
- Power Supply & Cable

**$14.98** Brand New

EXTRA PEN SETS ............................ $ .89 (Black) $3.98 (Color)

## 1200XL 64K COMPUTER

w/PAC-MAN **$49.** Reconditioned

## 1027 PRINTER

- 80 Column
- Direct Connect
- Letter Quality

**$79.** Brand New

## ATTN. DEALERS

PAC-MAN Case of 120 **$69.**

## 1050 DISK DRIVE

**$169.** Reconditioned

## 800 48K COMPUTER

w/PAC-MAN **$69.95** Reconditioned

## 1025 PRINTER

- 80 Column
- Dot Matrix
- Friction/Tractor
- Direct Connect

**$79.98** Reconditioned

## 800 BOARDS

- Mother
- CPU
- Power
- ROM

YOUR CHOICE **$8.98** Brand New

8Bit I/O Cable . $4.98

850 Interface Serial, parallel $79.98 Reconditioned

810 DRIVES $129. Reconditioned

BASIC TUTOR 4 BOOKS $4.98

5.25" DISKS 20 CENTS EA.*

| QTY. | PRICE |
|---|---|
| 10 | $4.00 |
| 100 | $29.95 |
| *1000 | $200 |

MAJORITY ARE UNNOTCHED CONTAINING OLD SOFTWARE

*NEW* DOS XE .. $9.95

DOS 2.5 w/manual .. $4.98

1010 Tape drive ...... $29.
10' Joystick ext. ...... $ .99
Trackball ................ $9.95

## CARTRIDGES FOR 800, XL, XE

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| BASIC CARTRIDGE | $4.95 | MATH ENCOUNTER | $7.98 | DEFENDER | $14.95 | BALLBLAZER | $19.98 |
| BASIC TUTOR (4 BOOKS) | $4.95 | DANCE FANTASY | $8.98 | ROBOTRON | $19.98 | BLUE MAX | $19.98 |
| TURMOIL | $4.95 | LOGIC LEVELS | $8.98 | TENNIS | $19.98 | STAR RAIDERS II | $19.98 |
| PAC-MAN (no box) | $4.95 | MEMORY MANOR | $8.98 | FINAL LEGACY | $19.98 | DAVID'S MIDNIGHT MAGIC | $19.98 |
| DONKEY KONG (no box) | $4.95 | LINKING LOGIC | $8.98 | MARIO BROS. | $19.98 | ARCHON | $19.98 |
| GORF (400,800) | $4.95 | CHICKEN | $8.98 | DONKEY KONG JR. | $19.98 | KARATEKA | $19.98 |
| DEMON ATTACK (400,800) | $4.95 | CLAIM JUMPER | $8.98 | JUNGLE HUNT | $19.98 | CHOPLIFTER | $19.98 |
| DELUXE INVADERS | $4.95 | DELTA DRAWING | $8.98 | MOON PATROL | $19.98 | GATO | $24.98 |
| JOURNEY TO THE PLANETS | $4.95 | HEY DIDDLE DIDDLE | $8.98 | BATTLEZONE | $19.98 | ACE OF ACES | $24.98 |
| STAR RAIDERS | $4.95 | SLIME (400/800) | $8.98 | FOOD FIGHT | $19.98 | LODE RUNNER | $24.98 |
| MISSLE COMMAND | $4.95 | ALPHABET ZOO | $8.98 | HARDBALL | $19.98 | BARNYARD BLASTER (gun req.) | $24.98 |
| ASTEROIDS | $4.95 | ALF | $8.98 | FIGHT NIGHT | $19.98 | DARK CHAMBERS | $29.98 |
| GALAXIAN | $4.95 | ADVENTURE CREATOR | $8.98 | ONE ON ONE BASKETBALL | $19.98 | AIRBALL | $29.98 |
| DEFENDER | $4.95 | DIG DUG | $14.95 | DESERT FALCON | $19.98 | SUMMER GAMES | $29.98 |
| E.T. | $4.95 | SKY WRITER | $14.95 | NECROMANCER | $19.98 | CROSSBOW (gun req.) | $29.98 |
| FACEMAKER | $4.95 | FOOTBALL | $14.95 | RESCUE ON FRACTALUS | $19.98 | CRIME BUSTERS (gun req.) | $29.98 |

## DISK SOFTWARE FOR 800, XL, XE

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| DAVID'S MIDNIGHT MAGIC | $4.98 | CRYSTAL RAIDER | $4.98 | PREPPIE II | $4.98 | PIRATE ADVENTURE | $4.98 |
| ZORRO | $4.98 | DESPATCH RIDER | $4.98 | THE COUNT | $4.98 | SECRET MISSION | $4.98 |
| BANDITS (48K 400,800) | $4.98 | MISSION ASTEROID | $4.98 | DISK 50 (50 GAMES) | $4.98 | VOODOO CASTLE | $4.98 |
| PROTECTOR II | $4.98 | SPIDERMAN | $4.98 | FREAKY FACTORY | $4.98 | TECHNICOLOR DREAM | $4.98 |
| CLAIM JUMPER | $4.98 | HULK | $4.98 | LASER HAWK | $4.98 | STRANGE ODYSSEY | $4.98 |
| SYNTREND | $4.98 | COMMBAT | $4.98 | ROCKET REPAIRMAN | $4.98 | VISICALC | $24.98 |
| CROSSCHECK | $4.98 | PREPPIE I | $4.98 | ADVENTURELAND | $4.98 | BOOKKEEPER W/ num. keypad | $29.98 |
| MOLECULE MAN | $4.98 | | | | | GET RICH | $29.98 |

## Astra

Double Density Dual Disk Drives (2 drives in 1 case)

**$199.** *New!*

## Special buys on 8-BIT Hardware!

## SAN JOSE COMPUTER

THE ATARI STORE

Sunrise Plaza 640 Blossom Hill Rd. San Jose, CA 95123
(408) 224-8575 • BBS (408) 224-9052

## Light Gun

For use with pistol games on 8-Bit systems

**$34.98**

### Light Gun Package

Includes:
- Light Gun
- Crime Busters
- Crossbow
- Barnyard Blaster

**$109.**

**SHIPPING:** ADD $5.00 TO ALL ORDERS. AIR AND INTERNATIONAL SHIPPING EXTRA. THAT'S IT.
**WARRANTY:** 90 DAY WARRANTY ON ALL ITEMS. **TAX:** CALIFORNIA RESIDENTS ADD 7% SALES TAX.
**PREPAYMENT:** USE VISA, MASTERCARD, MONEY ORDER, CASHIER'S CHECK OR PERSONAL CHECK.
PERSONAL CHECK MUST CLEAR PRIOR TO SHIPMENT. **C.O.D.:** CASH, CASHIER'S CHECK OR M.O. ONLY.

SUPER DISK BONUS

# Atar-I Ching

*The Computer of Changes foretells your fate.* **By Patrick Harvey**

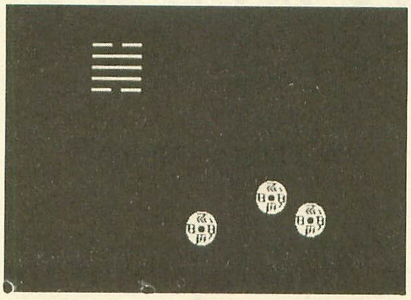
Let your faithful Atari "tell your fortune" and answer your questions about the future with this month's Super Disk Bonus, Atar-I Ching.

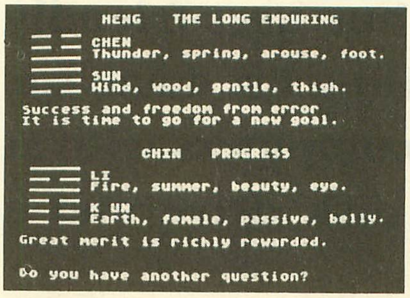
The I Ching is a most ancient Chinese fortune-telling system. In China its users have ranged from carnival fakes to generals and emperors. Its advice was used in everything from deciding what crop to plant to deciding military strategy.

Like many other traditional fortune-telling devices such as the Tarot cards, the I Ching is based on using "organized randomness" to try tapping into the underlying trends in the universe. Like most such devices, the key is in how you interpret the results, rather than the actual results themselves.

In one form of the I Ching, the user shakes numbered sticks in a container until one falls to the ground. Atar-I Ching simulates today's most widely used method—tossing three coins to create a set of lines. Depending on the number of coins that land with heads or tails up (ancient Chinese coins were inscribed on only one side and blank on the other) the line will be Yin, Yang, moving Yin or moving Yang. This is done six times to produce a "hexagram."

Yang lines are solid and Yin lines are broken. If there are any moving lines (all three coins are the same) a second hexagram will be drawn by changing the moving lines.






## ASKING QUESTIONS

Atar-I Ching will wait for you to mentally form your question and achieve the properly blank, receptive state of mind. The BASIC file is on this month's Antic Disk as ICHING.BAS and can be started by selecting it from the main menu. Then "toss the coins" by pressing the [SPACEBAR] *three times.*

Using the Atari's random number generator, the coins are tossed to determine a hexagram. Like binary code, six lines of either yin or yang provide 64 distinct hexagrams.

Based on the text associated with each of the hexagrams and component trigrams, plus your own intuitive understanding of the question, you can develop a forecast of future events. As the I Ching texts tend to be very vague, this can require a lot of imagination.

Atar-I Ching is a colorful, atmospheric program written by Patrick Harvey of Mountain View, California. This is his first **Antic** publication and he's a computer engineer doing customer training for National Semiconductor's 32-bit microprocessors.

Admittedly, Atar-I Ching is not quite as long as most of the other Super Disk Bonus programs that appear in **Antic**. However, the listing does include many lines crammed with interpretive text, which would be particularly time-consuming to type.

Your August 1989 Antic Disk—featuring Atar-I Ching as well as every type-in program from this issue—will be shipped to you within 24 hours after your order is received. Just phone Toll-Free to the Antic Disk Desk at **(800) 234-7001**. The monthly disk is only $5.95 (plus $2 for shipping and handling) on your Visa or MasterCard. Or mail a $5.95 check (plus $2 shipping and handling) to Antic Disk Desk, 544 Second Street, San Francisco, CA 94107.

# Customer List Manager

## *Much more than a phonebook*

By Thomas Andrews

**Keep your important lists up-to-date with this powerful but user-friendly BASIC telephone/mailing list database. Works on 8-bit Atari computers with at least 48K memory, disk drive and a printer.**

My family operates a retail vegetable business on our farm in central New York. We have a series of lists of customers who make quantity purchases each year and wish to be informed when certain seasonal products become available.

When we call these customers we want to have some information at hand concerning each one's preferences and a history of recent orders to aid them in making this year's purchases. Customer List Manager lets us find this information with ease. It's much better than the method we used before our Atari 800 arrived—a spiral notebook.

Customer List Manager is a telephone/mailing list/database program. Each record contains a customer's name, phone number and address, along with the type of product they want, some notes on personal preferences and a five year history of past orders. Each list has a title and year attached to identify it.

The list can be displayed on the screen, printed on most 80-column printers, or output to a word processor file. Entries can be edited or deleted, and histories updated, in one easy operation, thanks to the program's sorting and selecting routines.

The records in each list (referred to as "items" within the program) can be sorted on one or two levels. Subgroups of list records can be selected for processing under a search routine which allows use of a wildcard character in the search sequence. In case of disaster, the list can be reconstructed from the last editing operation or from the word processor file. All this—and it can be customized to better suit your own needs as well.

### GETTING STARTED

Type in Listing 1, MANAGER.BAS, and check it with TYPO II. Be sure to SAVE a copy before you RUN it. If you have trouble typing the special Atari characters in lines 50, 60, 65, 85, and 86, don't type them. Listing 2 will create these lines for you. Type in Listing 2, check it with TYPO II, and SAVE a copy.

When you RUN Listing 2 it will create a file containing these hard-to-type lines called LINES.LST. Merge this file into Listing 1 by typing LOAD "D:MANAGER.BAS" and then ENTER "D:LINES.LST". Be sure to SAVE the final version of the file.

If you have the Antic Monthly Disk, you should copy MANAGER.BAS to another disk before you try to use it,

since the program needs to be able to write files to disk. You can either copy the file from DOS using command C or O, or LOAD and SAVE the file from BASIC. Make sure the disk to which you copy MANAGER.BAS also contains DOS.

The program is menu driven and most operations are self-explanatory. When you first RUN the program the main menu will appear. If no list has been loaded, only options A (Start New List), C (Load List From Disk), and H (Change List Title) will operate.

Note that the [BREAK] key is NOT disabled. To return to the main menu after an accidental BREAK, type GOTO 100. Records in memory should remain intact!

## CUSTOMIZING LISTS

Customer List Manager uses a variable length field and record system to lower wasted memory space. Provisions have been made to limit the total list size to 19,000 characters and/or 100 records. This size allows the program and storage area to fit in a 48K Atari with standard DOS 2.5 configuration with about 1K bytes left over.

To change the maximum list length, change the value assigned to MAX in line 40. To change the maximum number of records, change the value of HUN in the same line. Remember that each record allowed uses seven bytes of bookkeeping overhead.

Each record has ten fields. The first five are labeled NAME, PHONE, ADDR., TYPE, and NOTES. These labels can be changed in line 90 by changing FNAME$. Be sure to use no more than five characters, padded with spaces if shorter, and separate each label with a colon.

The last five fields contain the history of the last five years and the labels for these fields are determined by the year of the list. The maximum length of any field is 60 and the minimum is 1. Anything outside these limits will be either cut off or padded with a space.

Both upper and lower case characters are allowed in the list. Inverse characters are not allowed and will be changed to normal if they are used.

Menu selections D, E, and G (editing, saving, and printing) allow the operation to be performed on only part of the items on the list. Items are flagged for processing through the use of a search routine that uses the question mark (?) as a wildcard character.

As an example, a search of phone numbers using the sequence 5?5 will flag all items with two fives separated by any other character in the phone number field. Use all capital letters in the search sequence, and Customer List Manager will search for both upper and lower case.

To change the wildcard character, change the question mark in the machine language routine SRCH in line 60 to any character you wish. Keep in mind that the character used as a wildcard cannot be used as a regular character. DON'T change anything else in line 60 as this could cause your computer to lock up. Also change the screen prompt in line 15060 as a reminder that a different wildcard is in effect.

## EDITING

The editing option is quite simple. There are three editing modes—edit and update, edit only, and update only. Update means that the list year is incremented by 1, the oldest year of history is dropped, the remaining history is shifted, and a new year is added to the end of each record.

The update-only mode is completely automatic and all list items will be updated. A space is inserted for the new history year field as the new list is written to a special edit file on the disk. After the entire list has been updated, the old list will be cleared from memory and the new list is input from the edit file.

The Edit and Update mode allows you to enter information for the newly-added year. You can also change the contents of any of the fields, or you can delete a whole item (or record) from the list. You must go through all of the items on the list in this mode. To edit, move the cursor to the desired line and change it to read as you like.

Because of the way the List Manager reads the list item from the screen, the INSERT and DELETE functions are disabled during editing, as is the clear screen command. When you have finished editing an item press [CONTROL] [A]. The item is read from the screen and placed in the edit file while Customer List Manager goes on to the next item.

To delete an item press [CONTROL] [D]. The item will not be sent to the edit file. (A last-chance "Are you sure?" question is asked before deletion is done.) If an item gets so messed up during editing that it would be easier to start over, press [ESC].

The edit-only mode allows the selection of part of the entries for processing. A menu comes up, letting you choose how much of your file you want to edit. If you decide to just edit some of the items (option B), you will be asked which field you want to use for the selection. Enter one, and you will then be prompted for the search sequence to use for a match. Remember, the ? character is the wild card that matches any other character.

If you want to leave edit-only mode without editing all the items flagged for processing, press [CONTROL] [M]. The rest of the flagged items will be moved to the edit file on disk, with no changes. The program will load the new list from the disk, ready for any further operations.

In all modes, once the editing operation is done, the edit file is left on the disk as a backup file that can be recovered using main menu option I. Each time a list is edited the old edit file is replaced with a new one.

List sorting can be done on any two fields. The primary field is selected first, followed by the secondary. To sort on only one field, use the same one for both primary and secondary.

The sort routine is completely automatic once started.

## PRINTING

The print option assumes a generic, no frills, 80-column printer using continuous paper. Therefore, it ought to work on just about any printer you would connect to your Atari. It is not necessary to print the entire list if you only need part of the pages. Customer List Manager will print any individual page or group of pages, and as with editing, you may select to print only those records which contain a specified string in the chosen field.

If your printer has capabilities beyond the minimum supported here (and most do) you might wish to write a new printer routine to take advantage of them. But if you don't want to do this, the program will print a list to a disk file that most word processors should be able to use. You can then use your word processor to add any special touches you want before you print out.

Customer List Manager has the ability to reconstruct a list from the word processor file (without the formatting commands or any other changes), providing a second emergency backup system. Use menu option I.

You can't merge two lists with Customer List Manager, but you can merge files using your word processor. Load the two list files into the word processor one after the other. (Do *not* mix Customer List Manager word processor files with files saved with menu option E. They use a different format.)

Remove the list title and year from the second list and save the completed list on your disk. Load the combined list into Customer List Manager using menu option I. If the combined list is longer than the maximum allowed, the program will load as many records as will fit, then stop.

It has been my experience that most practical application programs cannot precisely fit the needs of all users, and databases are no exception. That is why there are so many database programs out there—each user has different requirements.

Beyond changing field names and memory limits as discussed above, further changes would require more extensive rewriting of the program code. I have tried to program in a fairly modular, structured form to make this easier to do. If Customer List Manager does not suit your needs, study the listing and the instructions below, so you can adapt the database the way you want it.

## ABOUT THE PROGRAM

At the heart of Customer List Manager lies FIND, a machine language routine by Scott Sheck published in the December, 1984 **Antic**. The article, "Word Storage Space Saver," detailed an efficient method for storing strings of varying lengths.

As presented, FIND required that the strings be stored sequentially, unpadded with spaces, with the first character of each changed to inverse as a delimiter. The routine also required that an inverse character be appended to the end of the last field. Each string could be no longer than 255 characters.

FIND started at the beginning of a string and counted inverse characters until the desired field was encountered. FIND then determined the string position of the start and the length of the field.

This sequential access would be fine for most text adventures since FIND is quite fast. However, in the case of a search or sort operation where 100 or more calls to this routine are possible, or even likely, processing time mounts up quickly.

I added an index to the main storage area to provide pseudo-random access. The string position of the first character of each record is placed in a numerical array called LI. Now, if I want to examine the fifth field of record number 42, I use the contents of LI(42) as a starting point and go five fields from there. This provides very fast access—nearly as fast as if I had used fixed-length fields and calculated the position—and is much less wasteful of RAM.

Since the FIND routine uses inverse characters as delimiters, it follows that inverse characters cannot be allowed in the middle of a field. This necessitates a routine that does a character by character check of the input field, changing the first character to inverse and the rest of the characters to normal.

Since a BASIC routine to do this would be quite slow, I wrote the machine language routine CHNGR to do it more quickly. The call for CHNGR is:

```
DMY=USR(CHNGR,ADR(FIELD$),
LEN(FIELD$))
```

CHNGR is the address of the routine in memory that operates on the string FIELD$.

The USR routine SRCH will examine any block of memory for an occurrence of a specified sequence of characters. The sequence may contain any combination of letters, numbers, and punctuation. Capital letters (normal video) are checked as both upper and lower case. All other characters are examined only in the case and video type actually specified. The call for SRCH is:

```
X=USR(SRCH,ADR(SEQ$),
LEN(SEQ$),ADR(AREA$),
LEN(AREA$))
```

SRCH is the address of the routine that attempts to find SEQ$ in string AREA$. A one is returned in X if the sequence is found anywhere in the search area, and a zero if the search was unsuccessful. The length of the sequence may vary from 1 to 255 characters. The search area can be any length from 1K to 64K.

## SORT ROUTINE

The sort routine uses an insertion sort on two levels. For those unfamiliar with the insertion sort, consider a file drawer that needs sorting. You start on either end and thumb through the folders one at a time un-

til you find one that is out of the proper sequence. You then check back through the folders you have just done until you find the proper position for the misplaced one.

Then, you shift the folders to make room and insert the one in question in the right spot and go back to where you left off. When you reach the end, the sort is done. This type of sort, while certainly not the fastest type available, has the advantage of being very fast if most of the items are already in order, as is usually the case with Customer List Manager. It is also nearly as easy to program in BASIC as the bubble sort and is usually faster.

It is important to note that I do not actually move the text of the list around during the sort—I change the order of the pointers to that text. The reason for this is simple: When I move a pointer, I only have to move six bytes. Since the size of each record (item) in a list can vary anywhere from 10 to 600 bytes, not only would I have to move more bytes, but I would have to determine the length of the record I have to move each time. It is easy to see which way is better.

The usual method used during an insertion sort with a numerical array is to shift the array elements one at a time as you are checking for proper position. In this sort, I wait until I have found the proper position and shift the elements in a block using a machine language block memory mover, SHIFT.

This routine is a little different than the usual block mover in that it is bilateral. Most routines, including the string functions of Atari BASIC, will not work for moving in both directions when the source and destination blocks overlap. SHIFT checks for the direction of the move and starts at the correct end of the block to properly shift it in that direction. The call for SHIFT is as follows:

DMY=USR(SHIFT,HERE,THERE, LENGTH)

SHIFT is the address of the routine, HERE is the address of the first byte of the source, THERE is the address of the first byte of the destination and LENGTH is the number of bytes to move.

To determine the address of the numerical array, I DIMensioned a string just before it, determined the address of that string with the ADR function, and added the DIMensioned length of the string to it. Each element of the array takes six bytes. (Remember that the first element of the array has a subscript of 0.) Since I do not use LI(0) in CLM, I added six more bytes to the base address for LI.

---

*Thomas Andrews has a degree in electrical engineering. He found his Atari 800, some software and 12 issues of **Antic** at a garage sale in 1985.*

*Listing on page 34*

# VRename

## BASIC variable renaming tool.

By Doug White

**Clear up old BASIC programs and make your own programs more readable by renaming the variables with this handy BASIC utility. This BASIC program works on all 8-bit Atari computers of any memory size, with disk drive.**

Have you ever gotten a BASIC program out of an old magazine or from a friend that was difficult to read because it had cryptic names for all of the variables? While Atari BASIC allows variable names up to 120 characters long, some of the more primitive BASIC interpreters on some computers allow a maximum of two characters for each variable name. If you didn't write the program how can you tell what names like R9, D$, L2 and X$ mean?

Or have you used a subroutine from an old program in a new BASIC program that you were writing? There'd be a good chance that some of the variables were not compatible between that subroutine and the rest of the program. You had two choices—either change every occurance of each problem variable name throughout the entire program, or else add program lines to assign needed data from the variable names that the program uses to the corresponding variable names that the subroutine uses.

The first solution is time-consuming. The second creates confusing "spaghetti code," since the same piece of data will have a different name depending upon where it is in the program.

To help you (and me) write understandable programs and decipher programs already owned, I have constructed a programming utility that will let you rename any or all of the variables in a BASIC program.

This programming tool is simple in theory but powerful in practice. VRename can be used on completed programs stored on disk. But it is primarily designed to let you modify the variable names of a program that you are currently writing. For this reason the utility does not change the margins, colors, or any other feature of the display that you may have customized for your own convenience.

### GETTING STARTED

Type in Listing 1, VRENAME.LST, check it with TYPO II. This utility will not work with SAVE and LOAD—so LIST a copy of it to disk *before* you use it the first time. The utility deletes itself from memory after it is finished, so once you RUN it

you'll have to retype the whole program if you don't have a copy safe on disk.

Rather than making up a useless test program for demonstration purposes, I've included a program that contains a Shellsort routine in lines 500-680 for sorting numbers stored in an array named A() into descending order. Type in Listing 2, SHELSORT.BAS, check it with TYPO II, and SAVE a copy to disk.

The Shellsort is much faster than the commonly used bubble sort. If you want to use it in your programs, just rename A() to whatever name your program uses. If you need the numbers sorted in ascending order, change the < character in line 620 to a > character.

## USING VRENAME.LST

Now you can RUN the Shellsort program to try it out. Then, with the sorting program still in memory, ENTER "D:VRENAME.LST:" and RUN it.

The first variable used in the sort program, A(, will be shown on the screen. To rename it, type [Y] and press [RETURN]. Now type in the new name and press [RETURN]. Since the variable A( is an array, your new array name must also end with a ( character. Type [Y] and press [RETURN] if the name is spelled correctly.

The next variable name will now be shown on the screen. If you don't want to rename it, just press [RETURN] and the following variable name will appear. This process will continue until the last variable name of the sort program has been displayed.

Now the program will prompt you for a name for the disk file in which you want to keep the modified program. For example, type in D:SHEL2, or if you don't need a permanent copy you could use D8:SHEL2 to store it to a RAMdisk.

After the utility finishes, it will delete itself from memory. Press [RETURN] one more time to ENTER the modified program back into memory. Now if you LIST the program to the screen it contains the renamed variables. RUN the program and it will execute just as it did before.

## CHANGING YOUR OWN

VRename will work with almost any BASIC program, but your program *must* have line numbers greater than 0 and less than 31500. LOAD your program, then ENTER "D:VRENAME.LST" from disk. Then RUN the program, and rename variables as described above.

Be careful that you don't use reserved BASIC words (like LOAD, RUN, NEXT) as variable names. If you have a lot of variables, be sure you don't give different variables the same name by mistake. As always when modifying a program, it's a good idea to keep a backup copy on disk.

## HELPFUL HINTS

While you are typing in a program, you can use variable names that are short and easy to type. After you finish you can substitute longer, more descriptive names. Simpler variable names mean less typing, and will not only save you time but will probably reduce the number of typing errors that you make.

When your program is running properly, you can use my variable renaming utility to add descriptive names that use these characters without causing any problems.

## BASIC TOKENS

When you type in a program in Atari BASIC, the BASIC interpreter translates each program line into a line number and a series of string constants, floating-point constants, and tokens. Tokens are numbers that stand for each of the variables and BASIC commands in the program. The tokens are stored in a table called the statement table, which contains a "shorthand" version of the program. The tokenized version of the program in the statement table takes up less space and executes much faster than the program would if it were still in the original ASCII form.

If the value of a token is less than 128, it represents one of the BASIC commands or BASIC operators. If a token value is greater than 128 it stands for one of the variables in the program. For example, if the program contains a SETCOLOR command, the statement table will contain a 48 which is the token value for this command. In similar fashion a variable name will be replaced by a one-byte token each time it appears in the statement table.

The characteristics of each of the variables are stored in three other system tables— the variable name table (VNT) which contains the variable names that correspond to each of the variable tokens; the variable value table (VVT) which indicates whether a variable is a string, an array, or a simple variable called a scalar; or the string and array table (START) which holds the contents of every string variable and the value of each element in every array variable in the program.

In the variable renaming utility we are primarily concerned with the VNT. After a program is stored in memory, BASIC works with the tokens, the VVT and the START and ignores the variable names for the most part. Only when you are manually entering a program, LISTing it, or ENTERing it from disk or cassette are you directly working with the individual variable names. The SAVE and LOAD commands access the system tables only as a continuous block of data, and are of no use in renaming the variables.

All of the BASIC system tables that I've mentioned change size and move around in memory as different BASIC processes occur. The BASIC interpreter keeps track of the location of each of the tables using pointers.

Pointers are memory locations that contain the addresses of tables and other important data that move

around in memory. While the address stored in a pointer may change, the pointer itself stays in the same place. The variable renaming utility uses two pointers called VNTP, which points to the beginning of the VNT, and VNTD, which points to the byte immediately following the end of the VNT.

## HOW IT WORKS

When you ENTER the utility from disk, it merges itself around your program. Your program must therefore have line numbers greater than 0 and less than 31500. The utility jumps from line 0 directly to line 31500 and does not execute the original program. The utility then reads each variable name out of the VNT and asks if you want to change that name. The old variable name or the new variable name (if you just changed it) is then stored in VARNAME$.

The utility goes through all of the names in the VNT until it reaches VARNAME$, the first variable of the utility itself. (If no names were changed, the utility terminates and both the program and the utility remain in memory.)

Now there is a complete updated list of all of the variable names stored in VARNAME$. The utility puts the address of VARNAME$ into VNTP to make BASIC look for variable names in VARNAME$ instead of looking in the old table.

After you type in the name of the disk file where you want to store the modified version of your program, the utility LISTs lines 1-31499 (the original program) out to that disk file. As the program is being converted from tokens back into ASCII, the address in VNTP tells BASIC to read the updated names stored in VARNAME$.

Finally the utility clears memory with a NEW command, which terminates the utility. Press the [RETURN] key and the modified program will be reentered into memory from the ASCII file now on disk.

An added benefit of LISTing and ENTERing your program to and from disk is that it cleans up your program. Old unused variable names will be discarded. If your 800XL has Revision B BASIC ROM, 16 bytes of garbage are added to the end of your program each time you SAVE it to disk. Periodically LISTing the program to disk and ENTERing it removes this garbage. You have Revision B ROM if you PEEK(43234) and the result is 96.

## PROGRAM TAKE-APART

31514 VNT is assigned the address of the variable name table. VNT is incremented to keep track of the current character being read from the table.

31518-31530 Read each variable name.

31531-31598 Change a variable name and put the new name in VARNAME$. If unchanged the old name is put in VARNAME$.

31606-31614 Store the old variable name table pointers.

31618-31648 Make the variable name table pointers point to VARNAME$.

31652-31688 LIST the modified program out to a disk file.

31690 NEW clears out user memory and resets the system table pointers.

31694-31706 TRAP routine. If an error occurs while LISTing the modified program to disk, the old VNT pointers are restored.

## SPECIAL CASES

As I mentioned earlier, this utility is designed to merge itself around a program that is currently in memory. With a dimensioned size of 2000 characters, VARNAME$ will hold 128 names that average 15 characters in length, or 64 names that average 31 characters, or 32 names that average 63 characters, etc. If you have a particularly large program, both the program and the utility may not fit in available memory at the same time. If this occurs feel free to decrease the dimension of VARNAME$.

Another problem might arise with programs that use a lot of variable names. Atari BASIC limits you to 128 variables in one program. The renaming utility itself uses three string variables and 13 scalar variables.

128 – (3 + 13) = 112 VARIABLE NAMES

If your program uses more than 112 variables, ENTERing the utility may cause an error since the total number of variables exceeds 128.

First, try to localize the section of the program that contains the variables that you wish to rename. Rename the variables in that section. Then recombine the program section with the rest of the program.

If this process does not solve the problem, you will have to create a special version of the renaming utility which uses the same variable names as your program. Use the following procedure:

1. LIST your program to disk and type NEW to clear memory.
2. ENTER the utility into memory.
3. Delete line 31534.
4. Modify line 31680 to read: 31680 LIST NAME$
5. Type RUN to run the utility on itself.
6. As each variable name is displayed, rename it to one of the variable names in your program.
7. When the utility is finished, ENTER the modified copy of the utility into memory
8. Modify line 31680 to read: 31680 LIST NAME$,1,31499
9. LIST the modified utility back out to a disk file.

When you ENTER this specially tailored copy of the utility into memory with your program, BASIC will not have to add any new variables, since they already exist in your program. ▲

---

*Doug White of Arlington, Texas uses his 1200XL as an aid in designing and testing loudspeakers. His article* Equivalence *appeared in the February 1989* ***Antic.***

*Listing on page 39*

# Hi-Res PUT and GET

## Animation without Player/Missiles.

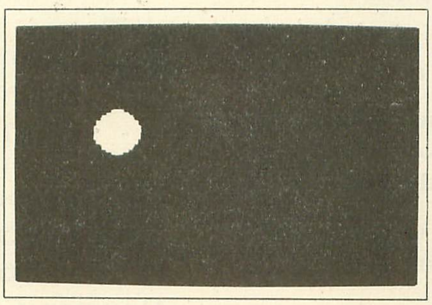
*By Brad Timmins*

**Here's a different way to create colorful objects that move smoothly around your computer screen. This BASIC demo program includes the handy PUTGET machine language routine for use in your own programs. It works on all 8-bit Atari computers of any memory size, with disk or cassette.**

Creating colorful objects that move smoothly around your computer screen can be one the of most satisfying and eye-popping effects your Atari has to offer. Player/Missile graphics were specifically designed for this purpose, but P/M graphics have their limitations. First, you can only have a maximum of five players (objects) on the screen at once. Each player can only be eight resolution-lines wide. Also, you are limited to one color per player.

Hi-Res PUT and GET overcomes these limitations by allowing you to take a "snapshot" of any rectangular section of the screen and store it in memory. Then you retrieve the image, and reposition it on screen.

For example, if you wanted to move a box smoothly across the screen, you would start by drawing the box using PLOT and DRAWTO commands. Then, using the PUT and GET routine, save to memory the portion of the screen where you drew the box. And finally, put the image back on the screen shifted over by one pixel from the previous image.



Cycling through this procedure will create the illusion of smooth motion. You can also cycle through changing images to give the illusion of animation.

### GETTING STARTED

To see the routine in action, type in Listing 1, PUTGET.BAS, and check it with TYPO II. Be sure to SAVE a copy before you RUN it. This demonstration program will draw an object and move it smoothly around the screen.

PUT and GET works with all high-resolution graphics modes (3-11). It also supports ANTIC modes E and D. ANTIC E is the so-called 7 1/2 graphics mode. This mode offers fine resolution and four colors to work with. ANTIC E is available on the XL/XE computers as Graphics 15.

ANTIC D has greater resolution than ANTIC E but only allows one color. This mode is available on XL/XE computers as Graphics 14. Atari 400/800 users must set up a special Display List to access these modes. PUT and GET does not support any text modes.

### USING THE ROUTINE

PUT and GET is written in machine language for maximum speed. It is called by the USR statememt:

X=USR(1536,XSTART,YSTART,
XWIDTH,YLENGTH,STORAGE,
GRAPHICS MODE,COMMAND)

XSTART and YSTART are the upper left corner of the rectangular section of screen you want to work with.

XWIDTH and YLENGTH define the size of the rectangle. XWIDTH defines the width of the rectangle. YLENGTH defines the length of the rectangle. Both can range from zero to the maximum coordinate value for the graphics mode you're in. For speed purposes, PUT and GET does not check for boundary errors, so be careful.

STORAGE is the area of memory you want to store your image in. To determine how much memory you'll need, multiply XWIDTH by YLENGTH, then divide the result by the appropriate number listed on the table below:

| Graphics mode | Divide by |
|---|---|
| 3 | 4 |
| 4 | 8 |
| 5 | 4 |
| 6 | 8 |
| 7 | 4 |
| 8 | 8 |
| 9-11 | 2 |
| 14 (ANTIC mode D) | 8 |
| 15 (ANTIC mode E) | 4 |

Add one to the result of the division if you get a remainder other than zero. If you are going to store your image in a string, add one to XWIDTH and YLENGTH before you multiply.

GRAPHICS MODE tells PUT and GET what graphics mode you'll be using. Values range from 3 to 13. Values 3 through 11 are the standard high-resolution graphics modes. Use a value of 12 for Graphics 14 (ANTIC D), and 13 for Graphics 15 (ANTIC E).

COMMAND tells PUT and GET what action you want done. If COMMAND equals zero, PUT and GET will copy a section of the screen to memory. If COMMAND equals one, an area of memory will be copied to the screen.

PUT and GET takes up most of Page six and all of the cassette buffer. So if you do any cassette operations, you'll have to reinitialize PUT and GET.

Machine language programmers can access PUT and GET with a JSR to $609, and by subsituting these Zero Page locations for the USR parameters:

XSTART = 224 (two bytes)
YSTART = 222
XWIDTH = 220 (two bytes)
YLENGTH = 218
STORAGE = 216
GRAPHICS MODE = 214
COMMAND = 212

Note that XSTART and XWIDTH should both be set up as two-byte values, even if the coordinate values are less than 255. ▲

*Brad Timmins is a freelance programmer living in Draper, Utah. His program* Macro RESET *appeared in the January, 1989* ***Antic.***

*Listing on page 38*

# Good King Zurp

The great pasta sauce chase. . . Don't ask! **Program by Frank Martone**




**Collect the ingredients you need to create a super spaghetti sauce in this crazy BASIC game. Good King Zurp will work on any 8-bit Atari computer with at least 24K memory, with disk or cassette, and your trusty joystick.**

Good King Zurp, son of Better King Xorpfgh, was a powerful, despotic figure in the land of Xjigqh, where they do everything the same as they do here except that they eat gelatin with chopsticks.

During the reign of Xorpfgh, the people of Xjigqh lived happy, prosperous lives, growing breadfruit and hiring themselves out as ballasts for the glowering Foon People across the Thplj River.

Good King Zurp ascended to the throne of Xjigqh when Better King Xorpfgh met an unfortunate end while stooping to pick a flower from a hostile grootlebush. Many suspected foul play, but everybody knew that Zurp couldn't possibly have masterminded any plan that required any type of native intelligence.

Zurp's first official act was to build—no, that's not right, his first official act was to legalize pouring beet syrup over your head and boiling grubs in mayonnaise and rutabaga oil. His *second* official act was to build a vast empire of financial wealth by hiring out his serfs as fruits and vegetables in traveling nutrional morality plays.

But unsurprisingly, hordes of toothpick craftsmen eventually overthrew Good King Zurp by disabling the palace guards, storming the king's rumpus room and overturning Zurp's pool table—with Zurp underneath, also thereby turning Zurp into a flapjack-like entity.

Zurp left Xjigqh reluctantly (and with constant neuralgia) for the sun-

nier pastures of Floobgrute, where he became the town's most popular rickshaw operator, but not before building his famous Horror Pits.

## NOW TO THE GAME

The leader of the insurgents, one Ftjorjt "The Mad Whacko Strange Person" Lipsko, also met an unfortunate demise when he competed in a polo championship but forgot his pony. His will stipulated that the new ruler of Xjigqh would be that man, woman, child or similar creature that could develop a nationally accepted spaghetti recipe.

As luck would have it, buried in Zurp's Horror Pits are hundreds of valuable things, mostly on the order of ingredients necessary for a really good meat sauce for pasta. Your job, would-be despot, is to root through these horrid, gloppy areas for gold nuggets in the form of packages of noodles, containers full of seasonings such as pepper, and cloves of garlic.

If you're lucky—that is, if you survive long enough to tromp through enough pits—you might chance upon some packages of really terrific tortellini, which almost assuredly would put you in the winner's circle.

This sounds easy enough, but this is no mere scavenger hunt, oh, no! No, you've got to avoid poisoned colanders (sometimes called "cullenders") and elude horrid, oversized maggots which will cover bits of you with an unpleasant substance—and if you're covered completely, you'll suffocate, which is to say the game will end rather abruptly and tell you that you failed miserably.

Each screen gives you 75 seconds to grab as much as you can before advancing to the next screen. Your score, the time remaining and the number of lives left—you start with nine—are all displayed at the top of the screen.

## GETTING STARTED

Type in Listing 1, PASTA.BAS, check it with TYPO II, and be sure to SAVE a copy to disk before you RUN it.

A title screen will appear, followed shortly by a list of objects and their point values. Press [START] to begin. Once the playing field is set up, simply move the joystick to direct your character towards the valuable items—but watch out for those maggots and colanders!

## VALUES

Here's how much each ingredient is worth:

| | |
|---|---|
| Packets of pasta (fettucini, spaghetti, vermicelli, sometimes gnocchi): | 100 |
| Packets of seasoning (including pepper, oregano and, for some reason, dried horseradish): | 250 |
| Garlic Cloves | 500 |
| Packets of special pasta (tortellini, tortelloni, ravioli, sometimes agnolotti): | 5,000 |
| Pesto sauce | 0 (there isn't any) |
| Maggots | you die |
| Colanders (cullenders) | you die |

## BONUS SCREENS

Every fourth screen is a bonus screen filled with rows of groaning colanders. Packets of pork-filled tortellini or spinach ravioli will appear randomly—and very briefly—onscreen. You must grab one before it disappears and pops up elsewhere.

If you've garnered over 60,000 points, first of all you should have a heck of a sauce recipe, but a nasty maggot will accompany you on the Bonus screens.

In order to get the bonus you must sucessfully grab three pasta packets without touching any colanders or maggots. If you fail, you go bonusless and advance to the next screen.

If you succeed, however, before heading on to the next screen you'll be given a random amount of points between 1,000 and 10,000.

The game gets tougher as your score increases. There'll be more colanders, more maggots and more spice packets and garlic cloves to grab. If you have over 50,000 points, the special pasta may appear on the playing field. If your score exceeds 100,000, the playfield will flash different colors, making it harder to concentrate.

## STRATEGY TIP

Maggots tend to dissolve colanders in both the regular playfield as well as in the bonus playfield. Nobody knows why. Fool them into passing over the colanders to dissolve them.

*Frank Martone is a student at Suffolk Community College on Long Island, New York. This is his first appearance in **Antic** and the editors apologize for the drastic rewrite of his original scenario.*

*Listing on page 40*

## COMPUTER SOFTWARE SERVICES

ATARI
P.O. BOX 17660, ROCHESTER, N.Y. 14617
PHONE (716) 467-9326

NOW "BEST BUY" RATED!

**JUST RELEASED!**

## $99.95 "SUPER ARCHIVER II"!

**(for ATARI 1050 drives)**

**NOW! COPYS** all **ENHANCED DENSITY** programs plus retains all of the features of our World Famous **SUPER ARCHIVER!** (see below). Allows you to **COPY** or **CREATE** single or **ENHANCED** density protection schemes (including **PHANTOM SECTORS!**). Completely automatic; compatible with the **BIT-WRITER!**; the **ULTIMATE BACKUP/PROGRAMMING** device! Only $99.95 plus $4 S/H/I
**NOTICE!** If you already own a **SUPER ARCHIVER!**, you can upgrade to a **SUPER** ARCHIVER II for only $29.95 plus $4 S/H/I (disk only - no additional hardware required).

## THE $69.95 "SUPER ARCHIVER"!® $69.95

**(for ATARI 1050 drives)**

The new **SUPER ARCHIVER**, obsoletes **all copying devices** currently available for the **ATARI 1050! It eliminates** the need for Patches, **PDB files**, Computer Hardware, etc. Copies are **exact duplicates** of originals and **will run on any drive**; without exaggeration, the **SUPER ARCHIVER** is the **most powerful PROGRAMMING/COPYING** device available for the 1050! Installation consists of a plug-in chip and 6 **simple** solder connections. Softwares included. Features are:

- **TRUE DOUBLE DENSITY**
- **ULTRA-SPEED** read/write
- **FULLY AUTOMATIC COPYING**
- **SUPPORTS EXTRA MEMORY**
- **SCREEN DUMP** to printer
- **TOGGLE HEX/DEC DISPLAY**
- **SECTOR** or**TRACK TRACING**
- **AUTOMATIC DIAGNOSTICS**
- **DISPLAYS HIDDEN PROTECTION**
- **ADJUSTABLE/CUSTOM SKEWING**
- **AUTOMATIC SPEED COMPENSATION**
- **AUTOMATIC/PROGRAMMABLE PHANTOM SECTOR MAKER**
- **ARCHIVER/HAPPY ARCHIVER COMPATIBLE**
- **BUILT-IN EDITOR** - reads, writes, displays up to 35 sectors/track short
- **BUILT-IN CUSTOM FORMATTER** up to 40 sectors/track
- **BUILT-IN DISASSEMBLER**
- **BUILT-IN MAPPER** - up to 42 sectors/track
- **DISPLAYS/COPIES** Double Density **HEADERS**
- **AUTOMATIC FORMAT LENGTH CORRECTION**
- **SIMPLE INSTALLATION**

The **SUPER ARCHIVER** is so **POWERFUL** that only programs we know that can't be copied are the newer **ELECTRONIC ARTS** and **SYNFILE/SYNCALC** (34 FULL sectors/track). If you want it **ALL** . . . buy the **"BIT-WRITER"!** also . . . then you'll **be able to copy** even these programs! Only $69.95 plus $4 S/H/I.

## $79.95 "BIT-WRITER"! $79.95

The Super Archiver **"BIT-WRITER"!** is capable of **duplicating even** the "uncopyable" EA and SYN series which employ 34 FULL sectors/tracks. **"BIT-WRITER"!** is capable of reproducing these and **FUTURE** protection schemes of non physicially damaged disks. **PLUG-IN** circuit boards and 4 **simple** solder connections. The **"SUPER ARCHIVER** with **"BIT-WRITER"!** is the ultimate **PROGRAMMING/COPYING** device for Atari 1050's. **EXACT DUPLICATES** of originals are made! Copies run on **ANY** drive. Must be used with Super Archiver. Only $79.95 plus $4 S/H/I.

## $69.95 "ULTRA SPEED PLUS" $69.95

Imagine a universal XL/XE Operating System so easy to use that anyone can operate it instantly, yet so versatile and powerful that every Hacker, Programmer and Ramdisk owner will wonder how they ever got along without it! Ultra Speed Plus puts unbelievable speed and convenience at your fingertips. Use **ANY DOS** to place an **ULTRA SPEED** format on your disks, boot any drive (1-9) upon power-up, format your **RAMDISK** in Double Density, activate a built-in **400/800 OS** for software compatibility, plus dozens of other features too numerous to mention! Below are just a **FEW** features you'll find in the amazing OS:

- **ULTRA** Speed SIO for most modified drives
- **ULTRA** Speed is toggleable
- Boot directly from **RAMDISK**
- Special timer circuits not required for 1 or 2 Meg upgrades
- Background colors adjustble
- Reverse use of **OPTION** key
- Cold-start without memory loss
- Built-in floppy/disk configuration editor (1-9)
- Built in **RAMDISK** configuration editor (1-9)
- **RAMDISK** exactly duplicates floppy drive so sector copying and sector editing are now possible
- Built in **MINI** Sector Copier
- Toggle **SCREEN OFF** for up to 40% increase of processing speed.
- Toggle internal **BASIC**
- Ram resident disk loader program (**MACH 10** menu)
- **DOUBLE DENSITY RAMDISK** capable
- Entire **MEMORY** test that pinpoints defective **RAM** chip
- Boot any drive (1-9) upon power-up or cold-start
- Supports memory upgrades up to TWO MEGABYTES
- **THREE** Operating Systems in one (**XL/XE, 400/800, ULTRA SPEED PLUS**)

## $29.95 "XF551 ENHANCER!" $29.95

The XF551 Atari drive is a fine product with one major flaw...it writes to side TWO of your flopping disks **BACKWARDS**. This causes read/write incompatibility problems with all other single sides drives made for Atari such as Indus. Trak, Rana, Percom, Astra, Atari 1050, Atari 810, etc. Add the **XF551 ENHANCER** to the new XF551 drive and your problems are over! This device will restore 100% compatibility while retaining original design qualities of Atari's super new drive. The **XF551 ENHANCER** is a **MUST** for all XF551 Owners. Installation is simple. Only $29.95 plus $4 S/H/I.

**DEALER/DISTRIBUTOR/USER GROUP** Discount availalbe call for info.
**PHONE Orders - MASTER CARD, VISA MAIL - Money Orders, Checks**

***SHIPPING! Add $4 for Shipping/Handling/Insurance within the U.S.A. UPS BLUE LABLE (2nd DAY AIR) available for $3 extra per shipment.**

CALL TODAY! (716) 467-9326
9 am - 5 pm (EST) WEEKDAYS

ATARI
COMPUTER SOFTWARE SERVICES
P.O. Box 17660
Rochester, N.Y. 14617

## $39.95 RICHMAN'S $39.95

**80 Column Word-Processor!**

Easy to use, very **powerful**, and **NO ADDITIONAL HARDWARE** required! Works with TV or Monitor! This **"DISK ONLY"** 80 Column Word-Processor is **simple to use** while offering numerous features such as:

- Completely Menu driven
- Self contained DOS
- Complete Text formatting
- Built in Printer Drivers
- Page numbering
- Justify and Endfill
- High Speed Cursor
- Dictionary
- 80 COLUMNS ON THE SCREEN!
- Insert and Delete
- Search Mode
- View only mode
- Memory Statistics
- Alternate Output
- Change screen Colors, borders, luminesce
- Mouse compatible

One expert has compared 40 **domestic** and **foreign word-processors** and **concluded that the RICHMANS 80 Column Word-Processor is the best!** Only $39.95 plus $4 S/H/I. (64K required).

## $39.95 THE "QUINTOPUS!" $39.95

The **"QUINTOPUS!"** is an inexpensive device that provides a tremendous amount of convenience while eliminating the problems associated with the endless "daisy-chaining" of peripherals (eg: drives, interfaces, printers, modems, cassettes, etc.). The **"QUINTOPUS!"** is an I/O port expander that converts a single I/O output/input into five additional outputs/inputs. Instead of daisy-chaining all of your peripherals (which often times results in passing a signal through 30 feet or more of cable!), the **"QUINTOPUS!"** allows each device to be connected directly to the computers through only three or four feet of cable. This is particularly useful when attempting to use ULTRASPEED or WARPSPEED data transfer rates. The **"QUINTOPUS!"** also provides the "extra" I/O ports often needed to connect devices not having daisy-chaining capabilities. Cable resistance and compacitance are greatly reduced thereby significantly improving the opportunity of accurate data transfers! Only $39.95 plus $4 S/H/I.

## $59.95 THE "QUINTOPUS!" $59.95

***(with SWITCHABLE PORTS!)***

This Deluxe version of the **"QUINTOPUS!"** has all of the above features with the additional benefits of **two SWITCHABLE PORTS!** This means you can connect two computers to one printer or two printers to one computer; you can switch in a computer/printer combination and while you're printing out a long document, switch in a second computer to access a modem, disk drive, cassette deck, etc! Switch multiple combinations of peripherals or computers without the hassel or re-arranging I/O cables . . . . simply flip a switch! Only $59.95 plus $4 S/H/I.

## LIMITED SPECIALS!

Previously we listed these products under our **FIRESALE** ad but many of our customers feared that the software items might be smoke or water damaged. They're NOT! All of the items listed below are BRAND NEW...only their literature or packing may have slight cosmetic imperfections. All items are fully warrantied! Quantities are limited! No rainchecks!

| | NORMALLY | LIMITED SPECIALS |
|---|---|---|
| 1. PILL (without case) | $ 70 | $ 19 |
| 2. SUPER PILL (without case) | 80 | 22 |
| 3. XL FIX ROM | 70 | 19 |
| 4. ULTRA MENU/DOS | 30 | 19 |
| 5. DISKCRACKER (Newest version) | 50 | 19 |
| 6. ELECTRONIC PHANTOM SECTOR MAKER DELUXE | 60 | 35 |
| 7. RICHMANS 80 COLUMN WORD PROCESSOR | 60 | 29 |
| 8. MIRACLE (Disk only Version of Impossible) | 70 | 19 |
| 9. IMPOSSIBLE for 800 or 800XL | 150 | 69 |
| 10. XL MATE | 30 | 15 |
| 11. COMPACTOR | 30 | 15 |
| 12. KLONE II (Generic HAPPY backup) | 100 | 75 |
| 13. SILENCER | 30 | 19 |
| 14. BLACK PATCH (MASTER) | 50 | 25 |
| 15. BLACK PATCH DATA DISKS 1 & 2 | 20 | 10 |

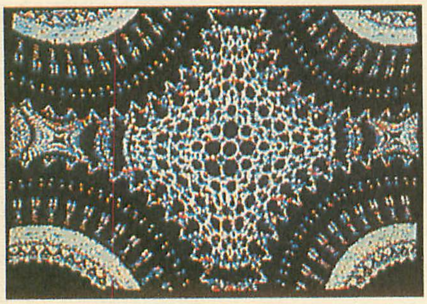
*Order 5 items or more and we will pay the freight!

These SPECIALS are for C.S.S. customers only - NO DEALER DISCOUNTS - NO RAINCHECKS - LIMITED QUANTITIES

# Superhop ACTION!

**Faster fractals grow in your Atari. By Douglas Skrecky**

***Watch elegant fractal patterns come alive onscreen at top speed with this fast and friendly ACTION! program. It works on 8-bit Atari computers with at least 48K memory memory and disk drive. If you are typing in the program, you will need the ACTION! cartridge from ICD. But if you own this month's Antic Disk, you will find a version that runs without the special cartridge.***



In May, 1987, **Antic** published my fractal generating program, *Dot Hopper*. That program used floating point arithmetic and was written in BASIC. Consequently, it was quite slow.

After playing with the ACTION! cartridge for awhile, I realized that floating point calculations were completely unnecessary for this language. Between ACTION!'s inherent speed and integer arithmetic, I found that the program could be made to run many times faster than in BASIC. Using ACTION!, the Atari could generate fractal patterns in seconds. The fractals appear to "come alive" and grow right before your eyes.

You don't have to know anything about the mathematics of fractal geometry to appreciate these elegant patterns. Some are startlingly organic, others are reminiscent of snowflakes and lace. If you enjoy the beauty of fractal imagery, you'll have fun with this program.

## GETTING STARTED

Type in Listing 1, SUPERHOP.ACT, and save a copy to disk before you start the program.

Antic Disk users don't need the ACTION! language cartridge to enjoy Superhop ACTION!. A runtime version, SUPERHOP.EXE, is on this month's disk. (As is usually the case, this runtime ACTION! translation would be too long to print as a type-in listing.) Copy SUPERHOP.EXE to another disk that has been formatted with DOS 2 or DOS 2.5, and make sure the disk contains a DOS.SYS file. Rename SUPERHOP.EXE to AUTORUN.SYS so that the program will load and run automatically.

To start growing fractals, turn off your Atari and insert your prepared Superhop ACTION! disk. Remove all cartridges (XL and XE owners press [OPTION]) and turn on your computer. Superhop ACTION! will load and run automatically.

When the program runs, it first presents the main menu, which lists

the various commands and options. You can return to this menu from any other screen simply by pressing [M].

Pressing [D] activates the demo mode, which generates successive fractals through 15000 iterations each (it doesn't take very long). Demo mode continues generating fractals until you press the [M] key to return to the menu, or press [C] to generate a single fractal.

While fractals are being generated in either Demo or Create mode, pressing the cursor keys (you don't have to press the [CONTROL] key with them) will shift the fractal so that you can view a different portion on the screen. However, the fractal is erased from the screen and the growth process must begin all over again—with the same pattern. The patterns are generated fast enough that this isn't a big inconvenience.

You may use the [>] key to "zoom in" on a portion of the fractal, enlarging it on the screen. The [<] key "zooms out". These keys also start the fractal growth process over again.

Pressing [C] changes the pattern. To see some of the possible patterns, try letting each pattern grow for only a few seconds before pressing [C] again. Then if a new pattern looks particularly interesting you can let it grow, and see how it develops.

Also, try letting patterns sit on the screen for several minutes after they seem to have stopped growing—they may surprise you.

## SAVE A FRACTAL

If you see a fractal pattern you like, you can save it to disk by pressing the [S] key. Doing so will read in a disk directory and let you choose a filename to save the screen under. The fractal is saved as a 62-sector Micro-Painter compatible file. Once the image has been saved, the program will return to the pattern in progress, which continues to grow without starting over.

To convert your Micro-Painter image to Micro Illustrator format, you can use *Rapid Graphics Converter* in the November 1985 **Antic**.

To load a fractal into Superhop ACTION! press [L]. A directory of the disk in drive 1 will be displayed. Enter the filename of the fractal you want, and it will appear on the screen. (Note that the fractal does not continue to grow after it has been loaded.)

Pressing the [SPACEBAR] will turn off the screen and speed up fractal growth by about 30%, although I realize that this is not an especially useful function!

---

*Douglas Skrecky is a mainframe programmer who currently lives in Vancouver, Canada. He likes to relax after work with his "trusty Atari 8-bit and perhaps a mug of beer."*

*Listing on page 37*

---

## *Coming Next in September 1989 Antic*

•

**Smart RAMdisk Handler**
automatically adjusts to your memory upgrade

•

**More from "Mapping The Atari"**
exclusive serialization of Ian Chadwick's reference classic

•

**Disk Bonus:**
*Electro-circuit CAD Artist*

# Mapping the Atari

Exclusive!

## **Antic** brings back the classic 8-bit reference book.

Serialization by Ian Chadwick

Ian Chadwick's "Mapping The Atari" has been one of the core references for Atari 8-bit programmers since the first edition was published in 1983. The book is set up as a comprehensive guide to the memory locations in the Atari 130XE, 65XE, 800XL, 600XL, 1200XL, 800 and 400. But it is much more than that. It is a virtual encyclopedia of indispensible information about the inner workings of the Atari.

This invaluable sourcebook essentially is out of print today, although a few copies of "Mapping The Atari" can still be found in the publisher's warehouse and occasionally at specialty computer stores. Therefore, it is with great satisfaction that **Antic** announces we are beginning an exclusive serialization of key excerpts from the revised second edition of Chadwick's book. This opening segment provides a useful overview of the material that will be covered in upcoming issues.

What exactly is a memory map? It is a guide to the memory locations in your computer. A memory location is one of 65536 storage places called bytes in which a number is stored. Each of these bytes holds a number for programs, data, color, sound, system operation, or is empty (i.e., has a zero in it), waiting for you to fill it with your own program.

Each byte is composed of eight bits, each of which can be either a one (on) or a zero (off). The alterable area of memory you use for your programs is called the Random Access Memory (RAM), while the area used by the Atari to run things is called the Read Only Memory (ROM). Although some of the memory locations in the special Atari chips were designed to be written to like the RAM, the rest of the ROM, including the Operating System ROM, cannot be altered by you since it contains routines such as the floating point mathematics package and the input/output routines.

I hope that you are familiar enough with your Atari to understand some of these rudimentary uses of a memory map. It is not the scope of this manual to fully explain how to use

PEEK and POKE statements. Briefly, however, PEEK allows you to look at the value stored in any one memory location. If you want that value to be printed to the screen, you must preface the PEEK statement with a PRINT statement such as:

PRINT PEEK (708)

If you haven't changed your color registers, this will return the number 40 to your screen. All bytes in the Atari can hold a number between zero and 255. POKE allows you to place a value into a byte, such as:

POKE 755,4

By doing this you will have turned your text upside down! You can return it to normal by:

POKE 755,2

Similarly, POKE 710,80 will turn your screen dark purple! As with PEEK, POKE can only involve numbers between zero and 255. You will not be able to POKE into most of the ROM locations since the numbers in many of them are "hard-wired," "burned" into the chip, and cannot be changed in this manner.

So how does the Atari (or other 8-bit microcomputers, for that matter) store a number larger than 255? By breaking it down into two parts; the **Most Significant Byte (MSB),** which is the number divided by 256 and rounded down to the nearest whole number, and the **Least Significant Byte (LSB),** which is the original number minus the MSB. The Atari knows to multiply the MSB by 256 and add the LSB to get the number. For example, the number 45290 is stored as two parts: 234 (LSB) and 176 (MSB). 176 times 256 equals 45056, plus 234 equals 45290.

## LEAST-MOST STORAGE

The Atari uses the convention of storing addresses in the LSB/MSB manner in memory (i.e., the smaller part is in the first memory location). For example, locations 88 and 89 store the lowest address of the screen memory. Let's say the numbers found there are 22 and 56, respectively. To get the decimal address, you take the MSB (stored in 89) and multiply it by 256, then you add it to the LSB at 88. In our case that's 56 * 256 equals 14336, plus 22 equals 14358. This is the address of the upper left corner of the screen. A simple way to do this in BASIC is:

BYTE = PEEK (88) + PEEK (89) * 256

The reverse (to break up a decimal location into MSB and LSB) is done by:

MSB = INT (BYTE/256):LSB = BYTE−MSB * 256

This process is easier for assembly language programmers who use hexadecimal numbers, since the right two digits are always the LSB and the two left of them are the MSB. For example:

$D016 (hexadecimal for 53270) equals 16 (LSB) and D0 (MSB)

$16 equals 22 in decimal, and $D0 equals 208 decimal. Multiply the MSB by 256 and add 22 and you get 53270. Throughout the map portion of this book I have provided both decimal and hexadecimal numbers together for ease of reference. In 8K BASIC, you can use decimal numbers only with POKE, and PEEK will return only decimal values to you.

Hexadecimal is a base 16 used instead of the normal base ten system because it is more suited to the eight-bit structure of the computer. So, when we say 2175 in decimal, what we really mean is:

| | | | | |
|---|---|---|---|---|
| 10000 | 1000 | 100 | 10 | 1 |
| 0 | 2 | 1 | 7 | 5 |

In hex, the same number is $87F. That breaks down to:

| | | | |
|---|---|---|---|
| 4096 | 256 | 16 | 1 |
| 0 | 8 | 7 | F |

Rather than multiply each next step up by ten, we multiply by 16. Okay, but where do we get "F" from? Well, if base ten has the numbers zero to nine, base 16 will have to have some letters added to the end to make up for the extra numbers:

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Decimal | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| Hex | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |

So $F equals 15 in decimal. Now here's how it all relates to binary math and bits:

Each byte can be broken up into two parts (nybbles), like this: 0000 0000

If each nybble is considered a separate number, in decimal, the value of each would range from zero to 15, or zero to $F. Aha! So if all the bits in each group are on (one, or set), then you have:

| | | |
|---|---|---|
| 1111 | 1111 | Binary |
| 15 | 15 | Decimal |
| F | F | Hex |

You join the two hex numbers together and you get $FF (255 in decimal), the largest number a byte can hold. So you can see how we translate bytes from binary to hex, by translating each nybble. For example:

| | | |
|---|---|---|
| 1001 | 1101 | Binary |
| 9 | 13 | Decimal |
| 9 | D | Hex |

$9D equals nine times 16 plus 13, or 157 in decimal.

| | | |
|---|---|---|
| 0100 | 0110 | Binary |
| 4 | 6 | Decimal |
| 4 | 6 | Hex |

$46 equals four times 16 plus six,

or 70 in decimal.

| | | |
|---|---|---|
| 1111 | 1010 | Binary |
| 15 | 10 | Decimal |
| F | A | Hex |

$FA equals 15 times 16 plus ten, or 250 in decimal.

Obviously, it is easier to do this with a translation program or a calculator!

Since I will often be discussing setting bits and explaining a small amount of bit architecture, you should be aware of the simple procedures by which you can turn on and off specific bits in any location (that is, how to manipulate one of the eight individual bits within a byte). Each byte is a collection of eight bits: numbers are represented by turning on the particular bits that add up to the number stored in that byte. Bits can be either zero (0 equals off) or one (1 equals on, or SET). The bits are numbered zero to seven and represent the following decimal numbers:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Value | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |

The relationship between the bits and the powers of two should be obvious. Adding up all the numbers (all the bits are set) gives us 255. So each byte can hold a number between zero (no bits are set) and 255 (all bits are set).

Sometimes, instead of zero, no bits set is intended to mean 256. That will be noted in the relevant locations. So how do you set a bit? Simple: POKE it with the appropriate number. For example, to set Bit 5, POKE the location with 32. To set Bits 7, 5 and 4, add up their values, 128 + 32 + 16, and POKE the location with the total: 176.

Sometimes you need to set a bit without changing other bits already set, so you:

POKE number, PEEK (number) + decimal value for the bit to be set.(i.e., POKE 50418, PEEK (50418) + 32)

To turn off a bit, instead of adding the value you would subtract it with POKE number, PEEK (number), minus the decimal value for the bit to be turned off. Binary math is simple and easy to learn; if you don't understand it now, you should do further reading on machine language before attempting any serious use of this guide.

## AND, OR, And EOR

It is useful to know how to perform Boolean logic on bits. There are three functions used in assembly code for bit manipulation in this manner: AND, OR and EOR (exclusive OR). Each requires you to use two numbers, the one being acted upon and the one used to perform the function. Here is a brief explanation of how these logical functions work:

AND is usually used as a mask—to zero out unwanted bits. You compare two binary numbers using AND; if both bits in the same location are one, then the result is one. If either bit is zero, then the result is zero. For example:

```
         51 = 00110011
AND      15 = 00001111
Result      = 00000011 = 3
```

OR is frequently used to force setting of a bit. If either bit in the original or the mask is one, then the result is one. For example:

```
         65 = 01000001
OR      128 = 10000000
Result      = 11000001 = 193
```

In this case, 65 is the ATASCII "A". By ORing it with 128, we get 193, the ATASCII inverse "A".

EOR "flips" bits in the original if the mask has a one in the same location. For example:

```
        193 = 11000001
EOR     128 = 10000000
Result      = 01000001 = 65
```

In this case, we have returned the inverse "A" to the normal ATASCII value. An EOR with 255 (all ones) will produce the complement of the number:

```
        171 = 10101011
EOR     255 = 11111111
Result      = 01010100 = 84
```

**In brief:**

| Original: | Mask: | AND: | OR: | EOR: |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 |

Atari BASIC supports AND, OR and NOT; NOT is the logical complement where NOT1 equals zero and NOT0 equals one. If the expression is true, you get a zero; if NOT true, a one is returned—for example NOT ((3 + 4) > = 6) results in zero.

In general, I have attempted to avoid using 6502 assembly language mnemonics, but have included them where I felt their use described the action to be taken better than a lengthy explanation. Most common are JMP (jump to location), JSR (jump to subroutine), RTS (return from subroutine), and RTI (return from interrupt). Readers should be minimally familiar with machine language in order to understand any machine language subroutines used here.

You can't hurt the machine by POKEing about in memory, although you may crash any program in memory, so SAVE your program first.

Usually you can salvage it by pushing [RESET], but you may have to turn off the machine and reboot on occasion. You can learn a lot about your machine by simply playing around with it.

## ABOUT LANGUAGES

The majority of the information here concerns language-independent locations and can be used regardless of the language you use for your programming. When the location is language-dependent, such as the BASIC or DOS areas, I have noted it in the proper section. You may exert the same control over your Atari in whatever language you chose. You will obviously have to change the commands PEEK and POKE to the proper commands of your language.

BASIC is a good language to start with: you can use it to learn programming, to explore your computer, to experiment with, and to have fun with. However, when you are ready to go on, you will have to learn a more efficient, faster language if you really want to make the best use of your Atari. Many people choose 6502 machine language because of its speed.

Computer languages, whichever you use, are quite exact in their meaning, especially compared to English. Consider that in English, a fat chance and a slim chance both mean the same thing. Yet POKE, PUT, and PUSH have very different meanings in computerese.

## GLOSSARY

**ANTIC, CTIA AND GTIA, PIA, POKEY**: Special Atari chips controlling the 400/800's graphics, color and screen resolution controller jacks and sound, respectively. Located in ROM, locations 53248 to 54783. ANTIC also processes the Non-Maskable Interrupts and POKEY processes the Interrupt Requests. These chips, along with the 6502 microprocessor which runs the rest of the Atari, are housed inside your computer, protected by the metal shielding underneath the plastic cover.

**BIT, BYTE:** A bit is the smallest size division of memory in your computer. It is so small that it can hold only one value in it: off (zero) or on (one). Eight bits together form a byte; this is the size of the memory locations discussed in this book. You will sometimes hear programmers talk about a half-byte called a "nybble."

**CIO:** Central Input/Output routines located in ROM. Controls Input/Output Control Block operations. Briefly, CIO handles the data input and output through the device driver(s) (also known as device handlers), then passes control to those drivers. It's a single interface with which to access all peripherals in a device-independent manner (i.e., uniform handling of data with no regard to the device being accessed). As an example: writing data to a disk file is treated in an identical manner as writing data to the screen; commas insert blanks between elements and both semicolons and commas suppress the End-Of-Line character (EOL).

**DCB:** Device Control Block, used by Serial Input/Output.

**DL:** Display List. This is a set of instructions which tell the ANTIC chip where to find the screen display data and how that data is to be placed on the TV screen.

**DLI:** Display List Interrupt. A DLI causes the display to stop processing to temporarily run a user-written routine.

**DOS:** Disk Operating System. The software loaded from disk file DOS.SYS that controls all disk I/O. The latest edition of DOS is called DOS 2.0S (S for single density).

**DUP:** Disk Utilities Package. The software loaded from disk file DUP.SYS that handles the DOS menu functions such as Copy.

**FMS:** (or sometimes DFMS): File Management System portion of DOS; a dedicated device driver that controls all I/O operations for device "D:".

**FP:** Floating Point mathematical package in ROM.

**I/O:** Input/Output.

**IOCB:** Input/Output Control Block. Area of RAM (locations 832 to 959) used by CIO to define operations to devices such as the disk drive (D:), printer (P:), screen display (S:), keyboard (K:) and screen editor (E:). ZIOCB is the page zero IOCB.

**IRQ:** Interrupt Request used for serial port communication, peripheral devices, timing and keyboard input. IRQ's are processed by the POKEY chip.

**NMI:** Non-Maskable Interrupt; used for video display and RESET. NMIs are processed by the ANTIC chip.

**OS:** Operating System. The resident system that runs the Atari.

Although people often refer to the entire ROM area as the OS, this is not correct. The OS ROM is that portion of memory which holds the floating point package, the Atari character set, the device handlers, and both CIO and SIO. The actual operating system itself is the portion of the OS ROM which handles the I/O.

**PMG, PM Graphics**: Player/missle graphics. Players and missiles are special moveable, user-defined, colored screen objects. They are often used for games, animation, or special cursors. PM graphics are unique in that you can establish the manner (priority) in which they interact with the rest of the screen display and each other.

**RAM:** Random Access Memory. All memory below the OS area (0 to 49151) which is used for storage, programs, buffers, cartridges, DOS, IOCB, shadow registers, and registers for the special Atari chips. Random Access means you can get to and from these locations at random, not that they store information randomly!

**ROM:** Read Only Memory. That part of high memory (locations 49152 to 65535) in which the special hardware chips and the OS reside. ROM is also used to describe cartridge memory such as the 8K BASIC ROM, which cannot be user-altered (the cartridge ROM supersedes the RAM). You

cannot alter most of the ROM, although some of the locations in the special Atari chips may be temporarily set to a new value.

With both RAM and ROM, we refer to areas with lesser values as being in "low" memory and locations with larger values as being in "high" memory.

**SIO:** Serial Input/Output routines located in ROM. Controls serial operations including the 850 interface (R:) and cassette recorder (C:). Briefly, SIO controls the Atari peripherals as per the request placed in its Device Control Block (DCB) by the proper device driver. It is also accessed by FMS for data transfer.

**VBI:** VBLANK interrupt. A VBI is an interrupt that occurs during the VBLANK interval, causing the computer to jump to a user-specified location to process a short user-written routine during the VBLANK process.

**VBLANK:** Vertical Blank. The interval between the time the TV electron beam turns off after reaching the bottom right corner of the screen and returns to the top left corner and turns back on again. This small time period may be used by machine language programmers for short routines without interrupting the display by writing a VBI (above). There are two VBLANK stages. Stage one is performed every VBLANK cycle (1/60 second). Stage two is performed either every 1/30 second or every 1/60 second when it doesn't interrupt time-critical code being executed. See the end of the memory map for the processes executed in each stage. ▲



MAPPING THE ATARI
$16.95, COMPUTE! Books, P.O. Box 5406, Greensboro, North Carolina 27403, (919) 275-9809.

*Ian Chadwick is a Toronto-based freelance writer.*

## A Graphics Memory Map

This diagram is not to scale; it is merely meant to give you a visual idea of the structure of the Atari memory. The numbers on the right are the memory pointers: these locations point to the addresses shown. The numbers on the left are the actual locations in memory.

## Notes

The bottom of the BASIC RAM depends on whether or not you have DOS files loaded in. Without DOS, LOMEM should be 1792, with DOS 7420. If you increase or decrease the number of disk and sector buffers by modifying DOS, this value will change again. See locations 743, 744 and 1801, 1802.

The size and location of the variable, string and array tables depend on the program use and size. The more variables and arrays, the larger the memory the tables use.

The size and address of the Display List and screen memory depend on the GRAPHICS mode in use.

The first 256 bytes pointed to by LOMEM are the token output buffer. The actual BASIC program starts at the address pointed to by VNTP.

## Atari Timing Values

| | | |
|---|---|---|
| clock frequency | = | 1.79 MHz |
| 1 machine cycle | = | 0.558 µsec. |
| 1 frame | = | 1/60 second |
| scan lines | = | 262/frame |
| color clocks | = | 228/scan line |
| color clocks | = | 2/machine cycle |
| machine cycles | = | 29868/frame |
| machine cycles | = | 114/scan line |

VBLANK time = 7980 machine cycles or less, depending on GRAPHICS mode. The shortest 6502 instruction requires two cycles; during that time the electron beam moves four color clocks.

## Horizontal blank time:

| | |
|---|---|
| Wide playfield | 18 machine cycles |
| Normal playfield | 34 machine cycles |
| Narrow playfield | 50 machine cycles |

| Location | Contents | Pointers |
|---|---|---|
| 65535 | Top of memory<br>Operating System ROM | |
| 60906-65535 | Device handler routines | 794-831 HATABS |
| 59716-60905 | Serial Input/Output (SIO) utilities | |
| 59093-59715 | Interrupt handler | 512,513 VDSLST<br>514-527 Vectors |
| 58534-59092 | Central Input/Output (CIO) utilities | |
| 58533 | Operating System vectors | |
| 58496-58533 | Initial RAM vectors on powerup | |
| 58448-58495 | JMP vectors | |
| 58432-58447 | Cassette | |
| 58416-58431 | Printer | |
| 58400-58415 | Keyboard | |
| 58384-58399 | Screen | |
| 58368-58383 | Editor | |
| 58367 | ROM Character set | 756 CHBAS |
| 57344 | | |
| 57343 | Floating Point ROM package | |
| 55295 | I/O chips | |

| Location | Contents | Pointers |
|---|---|---|
| 54784-55295 | Unused | |
| 54272-54783 | ANTIC | 756 CHBAS |
| | | 755 CH1 |
| | | 564-565 LPEN |
| | | 560-561 SDLSTL |
| | | 559 SDMCTL |
| 54016-54271 | PIA | 636-639 PTRIG# |
| | | 632-635 STICK# |
| 53760-54015 | POKEY | 624-631 PADDL# |
| | | 562 SSKCTL |
| | | 16 POKMSK |
| 53504-53759 | unused | |
| 53248-53503 | GTIA or CTIA | 704-707 PCOLR# |
| | | 708-712 COLOR# |
| | | 644-647 STRIG# |
| | | 623 GPRIOR |
| 53247 | Unused 4K ROM block | |
| 49151 | 8K BASIC ROM<br>or Left cartridge (A) | |
| 40959 | Top of BASIC RAM or | 106 RAMTOP |
| | | 740 RAMSIZ |
| | Right cartridge (B) ROM if present<br>(Atari 800 only) | |
| Size and location vary with GRAPHICS mode | | |
| | Text window screen RAM<br>40800 for GR.0 | 60,661 TXTMSC |
| | Bottom of screen RAM<br>40000 for GR.0 | 88,89 SAVMSC |
| | Display List:<br>39968 for GR.0 | 560,561 SDLSTL |
| (OS) | Top of BASIC RAM | 741,742 MEMTOP |
| 32768 | | |
| 32767 | User-program RAM | |
| | The amount of RAM can be ascertained by:<br>PRINT FRE(0) | |
| (13062) | Bottom varies: see note below<br>Depends on buffer area allocated. | |
| | RAM used by DOS and File System Manager | |
| | Stack for FOR-NEXT & GOSUB | 144,145 MEMTOP<br>142,143 RUNSTK<br>14,15 APPMHI |
| Size and location vary with program size | | |
| | String & array table &<br>end of BASIC program | 140,141 STARP |

| Location | Contents | Pointers |
|---|---|---|
| | BASIC program area | |
| | Statement table:<br>Beginning of BASIC program | 136,137 STMTAB |
| | Variable variable table | 134,135 VVTP |
| | VNTP + 1 | 132,133 VNTD |
| | Variable name table | 130,131 VNTP |
| (7420) | BASIC bottom of memory | 743,744 MEMLO<br>128,129 LOMEM |
| | Sector buffers | 4921,4937 SABUFL/H |
| 6781 | Drive & sector buffers | 4905,4913 DBUFA1/H |
| 6047 | DOS vector | 10, 11 DOSVEC |
| 5440 | DUP.SYS start | |
| 5377 | VTOC buffer | |
| | DOS initialization | 12,13 DOSINI |
| | or BASIC RAM without | (743,744 MEMLO) |
| | DOS resident | (128,129 LOMEM) |
| | FMS RAM | |
| 1792 | DUP.SYS beginning | |
| 1791 | RAM used by OS and cartridge.<br>(to bottom of RAM) | |
| | Page six RAM | |
| 1535 | RAM used by BASIC<br>(to bottom of RAM) | |
| 1406 | Floating Point RAM | |
| 1405 | BASIC RAM | |
| 1151 | Operating System RAM | |
| | Cassette buffer | |
| | Printer buffer | |
| | IOCB's | |
| 512 | | |
| 511 | Stack | |
| 256 | | |
| 255 | BASIC zero page RAM | |
| | Floating Point pg. 0 | |
| | Assembler Cart. pg. 0 | |
| 128 | | |
| 127 | OS page zero RAM | |
| | Zero page IOCB | |
| 0 | Bottom of memory | |

# Carnival of Cartridges

*Food Fight, Karateka and lots more bits.*

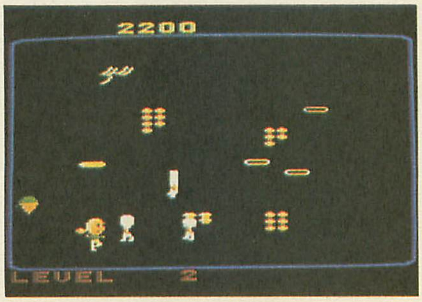
**Reviewed by David Plotkin**

## FOOD FIGHT

**Food Fight** ($24.95) is a very simple (but messy) game. You control Charley, who starts out on the right side of the screen. The object is to guide him to the left side, where a dripping ice cream cone awaits him. Charley has 32 seconds to get across the screen, otherwise the ice cream melts, costing him one of his three lives. Sounds easy, right?

Well, of course, there is more to it than that! There are manholes strewn across the screen, and you must take care that Charley doesn't fall into one. More importantly, cooks emerge from these manholes, and they chase Charley across the screen, trying to keep the poor boy from his just desserts. If one of these cooks touch Charley, there goes another life.

Fortunately, Charley is not defenseless. There are piles of food on the screen, and he can grab these (one pile at a time) and toss them at the cooks. If he hits a cook with a banana, tomato, pie, etc., the cook becomes so embarrassed he turns red and leaves the screen. Unfortunately, another soon replaces him.



*Food Fight*

Food Fight is a challenging (and frustrating) game because the placement of the food on the screen appears to be random. Sometimes the cooks start out closer to poor Charley than a pile of food does. The cooks can run faster than Charley, so unless he can get to a pile of food and load up with some throwing weapons, it's curtains for him.

Still, the action is fast and the graphics are good. You'll want to use a good joystick for this one.

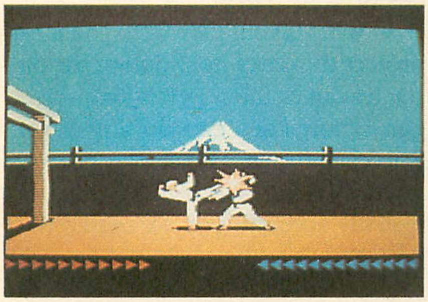
## KARATEKA

In **Karateka** you are an expert in the martial art of Karate. Your village has been destroyed and your intended carried off by the evil warlord, Akuma. You must defeat Akuma's warriors in battle to rescue your sweetheart.

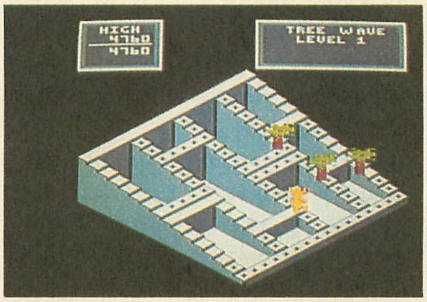
You can control the blond hero of Karateka with keyboard or joystick. Using the joystick, pressing the button quickly unleashes a punch, while holding down the button results in a kick. In the heat of battle it's hard to control whether you punch or kick. Blows can be directed high, medium or low by moving the joystick in the appropriate direction before pressing the button.

The keyboard controls work quite well. The arrow keys direct your alter-ego forward and back, while an easily-reached set of six keys direct punches and kicks. Because each key is assigned its own function, you can achieve very precise control, which is certainly necessary in the upper levels of the game.

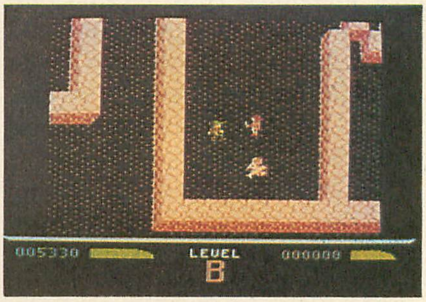
## Atari's latest release of XE game cartridges—which run on all Atari XL/XE computers with at least 64K memory—revives a number of coin arcade or disk classics that had been unavailable in the Atari 8-bit market for some time.



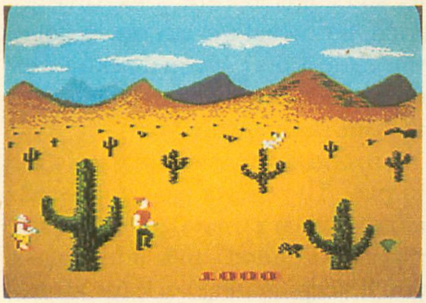
*Karateka*




*Crystal Castles*



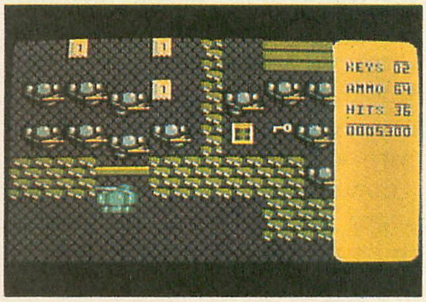

*Dark Chambers*



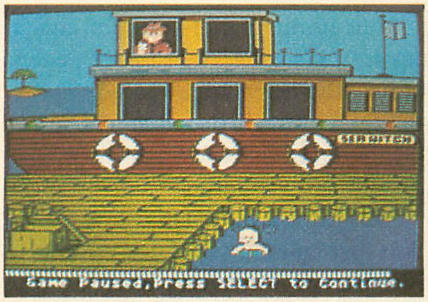
*Crossbow*




*Choplifter*



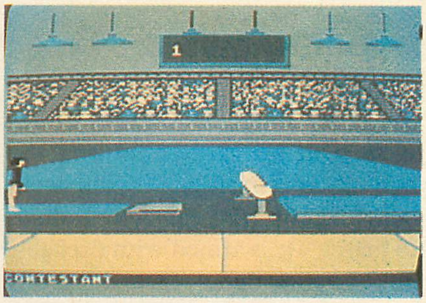

*Into The Eagle's Nest*




*Crime Buster*




*Airball*




*Summer Games*

As you defeat opponents, you begin moving towards the castle. Inside the castle, you will have to face more opponents, generally far more skillful than those you faced outside. You also have to figure out how to get past the diabolically deadly iron gate. And Akuma's war bird will attack to steal your strength.

Karateka is a multi-faceted game, requiring considerable arcade skill, plus the ability to recognize the attack patterns of your opponents. The graphics and animation are superb. The smoothly scrolling high-resolution background is extremely realistic, and the karate moves of both your hero and the opponents are true to life, right down to the swishing of the robes. And if you ever manage to see the final sequence (remember to bow to your intended. . .) it will make all the hours spent with this game very worthwhile.

### CRYSTAL CASTLES

In **Crystal Castles** ($24.95), you

guide Bentley the bear as he makes his way through a variety of three-dimensional mazes. The paths in these mazes are littered with gems, and Bentley must pick them all up before he can move to the next level. He can also pick up pots of honey for extra points.

A whole host of creatures try to prevent Bentley from achieving his life's ambitions. These creatures include animated trees, skeletons, and a swarm of bees. Bentley can grab a magic hat to protect him for awhile. He can also jump over many of his enemies.

The mazes in Crystal Castles are what set this game apart from others—previously as a coin-op hit and now as an XE game cartridge. In extra-sharp colors, the mazes simulate 3-D, with gem-laden paths on multiple levels connected by elevators. There are also tunnels, and when Bentley goes into a tunnel or behind a structure, he becomes an outline so that you can see where he is.

Controlling the game takes a little getting used to, because the joystick diagonals are used to guide Bentley through the maze. But the system can be mastered and the game is a lot of fun.

## DARK CHAMBERS

**Dark Chambers** ($34.95) is one of the best dungeon arcade games released in XE cartridge. Using the joystick, you must guide your well-animated onscreen character though 26 mazes. He is armed with fireballs, and it's a good thing—these dungeons are full of dangerous beings.

At various places in each level "spawners" create the enemies, which are set up in levels of power. The most powerful is a Grim Reaper. Shooting one of these turns it into a Wizard. Shooting a Wizard turns it into a Wraith, then a Skeleton, and finally a Zombie. Shooting a Zombie kills it. Thus, some of your enemies must be shot five times to kill them!

The spawners keep creating creatures until you destroy them by shooting them. As you have probably have guessed, it takes five shots to destroy a Grim Reaper spawner (through Wizard spawner, Wraith spawner, etc.).

Fortunately, the enemies aren't very bright (But then, you aren't either, or you wouldn't be in this dungeon.) So they simply line up to be killed. You lose life energy if an enemy does touch you, and the game is over when your life energy is gone. You can recharge your energy by eating food and drinking potions which are found in the dungeon.

Other things can be found in the dungeon, such as the Keys necessary to open locked sections. Bombs which destroy all enemies on the screen, treasure, or extra weapons such as shields and more powerful fireballs are also to be found. But look out for the traps and poison, which all cost you life energy.

The smooth animation and fast action, coupled with the excellent playability (at the beginner setting, even a novice should be able to survive quite a while) make this a superb game. The 26 levels provide lots of variations as well.

## CROSSBOW

**Crossbow** ($34.95) is a game for the Atari light gun. The opening screen shows a large number of locations including a desert, volcano, castle, town and jungle. From your current location, two or three paths are available, and these paths lead to other locales. Unfortunately, the paths don't show on the screen, so you'll probably want to map where the three color paths lead to so that you don't wander aimlessly around the screen!

Once you choose a path, the characters in your party (referred to as your "friends") set out on their journey. The scene switches to the selected location, and your party appears, moving leisurely across the screen while all heck breaks loose around them.

Large numbers of unfriendly creatures try to wreak havoc on your friends. Lightning bolts descend, ghosts appear, rocks careen down the path, attacks come from spiders, scorpions, alligators, pterodactyls (!) and man-eating plants—to name just a few of the different hazards in each location. Of course, if anything touches one of your friends, the poor soul is fried and no longer remains a member of your party.

So what is your job? Well, you must protect the party by shooting the obstacles out of the way with your light gun before they can do any harm. This can be quite a challenge, because not only is the Atari Light Gun not especially accurate, you are required to protect four or five characters from all those hazards.

The game becomes a little easier after some members of your party get fried, but when the last one is gone, the game is over. If you do extra well on a screen, another person is added to your party (lucky you!).

Crossbow, I must admit, is just a gas to play. The effects are great (watch closely when a friend gets fried), the playability good and addicting—you'll keep coming back for more.

## THUNDERFOX

Your home planet is under attack (again!). It's up to you in your Federation **Thunderfox** ($24.95) space fighter to destroy the alien transports bringing special crystals to power a deadly war machine.

This game is played in stages. When you first launch, you must approach the transport, dealing with formations of enemy fighters that try to both ram you and fire at you. Additionally, the transport has powerful guns, and part of the transport sticks up so high that you will crash if you try to fly over these portions of the ship.

Your fighter is armed with two weapons—guns which are effective against enemy fighters, and bombs which must be used against the transport. The transport has "ground in-

stallations" which are good for points but should be ignored because they are surrounded by obstacles you can crash into.

The object is to bomb the two "anti-gravity stabilizers". Each one must be hit five times. Since you can't carry ten bombs, you must return to your mother ship to reload, then return to finish the job.

If you can manage to disable the stabilizers, a door in the transport opens and you enter the thermonuclear laser room. You must maneuver through some very close quarters and avoid enemy lasers—hurrying all the while because of the room's radiation. If you survive, the third level has you facing off against a large "Crystal Guardian," which takes multiple hits to kill.

Thunderfox is simply too hard. Typical games last only seconds, especially since your fighter "bounces" off the invisible boundaries of the scrolling screen—usually into an obstacle. You can have just a single missile on the screen at once, which further limits your shooting.

## CHOPLIFTER

The 64 delegates to the Peace Conference have been taken hostage by the Bungeling Empire and placed in locked bungalows. Your mission in **Choplifter** ($24.95) is to pilot a helicopter from a secret base (disguised as a post office) into enemy territory and rescue the hostages.

Your chopper can hold sixteen hostages, which means you must make multiple trips. Each trip gets harder because the enemy adds more sophisticated weapons to make your life miserable.

On the first level you face tanks, which are rather easily avoided by simply staying above their fire and blasting them into tiny pieces with your guns. The danger of the tanks, however, is that the gunfire will kill some of the hostages, so it is best to take out tanks as soon as possible.

You must blast open each bungalow. (Or let a tank do it for you, they're really good at it.) Little figures will then run out and wave at you. Be careful where you set down your chopper, or you might crush a few hostages. When you land, the figures run to the helicopter and climb aboard. You can then fly back to base and unload for the next trip.

Starting with the second trip, more enemies show up—jets firing missiles and homing drones. The drones are rather easily destroyed, but the jets are tough—they come in fast and are hard to hit.

The helicopter can face forward, left or right. To switch directions, you must press the fire button and pull the stick in the direction you want to turn. Unfortunately, this is exactly the same maneuver you make in the heat of battle when you *don't* want to turn! Thus, you end up changing directions a lot when you don't want to, which can shorten your games considerably.

The graphics are incredible. The helicopter, enemies, background and base are highly detailed, and the rescued hostages even wave to you as they get off the chopper. Choplifter is a timeless classic on the Apple and Atari 8-bit computers, and the even more detailed graphics in this version help make it a very good game.

## INTO THE EAGLE'S NEST

Three of your comrades have been captured in **Into the Eagle's Nest** ($24.95). The Eagle's Nest is a Nazi fortress where top officials are meeting to plan a counterattack. You must try to rescue the three saboteurs and blow up the castle with the explosives they planted. The castle contains locked doors, many floors, an elevator, chests, jewels, spare ammunition and many, many soldiers.

You must negotiate the castle (which looks suspiciously like a maze), blasting Nazi soldiers as you go. They, of course, shoot back (or more accurately, try to hit you), and if you are hit 50 times, you die and the game ends. You can recover from these hits by finding medical kits and food which have been rather carelessly left about the castle.

Pay close attention to ammunition too, because you can only carry 99 rounds, and you run out surprisingly quickly. If you manage to find your comrades, you must attempt to exit the castle with them. This is tough, because they are weak and sick and don't move too fast.

The Nazi soldiers (This game should sell real well in Germany!) simply line up to be shot, much like the creatures in Dark Chambers. In fact, this whole game is quite similar to Dark Chambers, but not as playable. Your lack of ammunition makes it tough, and a single mistake (like shooting a box full of dynamite) can cost you the game! Overall, Dark Chambers is a better game of this same type.

## CRIME BUSTER

**Crime Buster** is an exciting game that uses the Atari Light Gun and provides a "blasting" good time. You play one of the city's top detectives, out to wipe out crime once and for all.

You choose a one- or two-player game by blasting holes in the appropriate choice. You can, of course, shoot holes in anything you want. The destructive effect is quite gratifying.

You must choose a section of the city from a map—just fire twice at the section you want. If the section of the city you choose is *not* adjacent to the section you currently occupy, you must travel to the portion of the city in question. Travel is very dangerous, and the best strategy is to avoid it. If you do have to travel, your car appears on the screen, traveling down a road which scrolls from left to right.

You control the car by firing at arrows pointing left and right, which makes your auto move towards that direction. Periodically a gangster car appears and fires at you. If it hits you, your car explodes and you lose one of your three lives.

To fire back, you must shoot your light gun at arrows which determine your direction of fire. This is difficult, since you can't control the car's position and fire at the mobsters at the same time. It would have made more sense to let you fire at the mobster's car with your light gun. If you somehow manage to hit the mobster's car, it rolls over and explodes.

At your destination, your goal is to plug all the gangsters in a classic shoot-out, while at the same time avoiding the innocent bystanders who don't have the good sense to get out of the way. Complicating matters, the gangsters sometimes dress up like innocent bystanders to try and fool you into not shooting them.

Each scene is different. Settings include buildings, a ship, and the inside of a warehouse. Gangsters appear in doorways and windows, pop out of boxes and manholes, and even come up from the water.

The graphics in Crime Buster are very good. The mobsters' cars rolling over and over are very satisfying. When you blast the gangster with the hat on, his face changes and the hat flies off. Other effects are equally good. Overall, Crime Buster is an excellent game which will keep you coming back to try to do better.

## AIRBALL

In one of the more bizarre scenarios for video games, you become the plaything of an evil wizard who turns you into an **Airball**—with a slow leak! In order to escape this fate, you must find a spellbook and return it to the wizard. Of course, the spellbook is hidden in a 150-room mansion, which also contains spikes, needles and other nasty, pointed objects detrimental to the health of an airball.

You move from room to room using your joystick and make your airball hop into the air by pressing the fire button. The rooms are viewed in three-quarter perspective, which can be confusing.

Bonus objects in the mansion include gold bricks and precious stones. There are also crates which the airball can move, to uncover valuable objects. You may also find such useful objects as a lantern, which can be quite handy in the rooms that have no light! When your pressure gets low, you must try to find an air pump—and then avoid death by overinflation.

The graphics in Airball are very well done and detailed, with many of the rooms containing elaborate decorations. Strategy definitely plays a significant part in winning this game, as it is quite easy to get lost in the huge mansion. Be warned, though. This is an extremely frustrating game, with death just the barest miscalculation away.

## SUMMER GAMES

**Summer Games** presents eight Olympic events for you to compete in, including diving, skeet shooting, pole vault, two swimming events, gymnastics and track. You may choose to compete in all the events in order, compete in a single event, or practice any given event.

If you decide to compete, you must choose a name and country. Just highlight the flag of the country you want and that nation's national anthem will play. The impressive opening ceremony has the Olympic theme playing, a runner lighting the torch, and doves flying away.

The events themselves basically test your skill and timing with a joystick. Some work pretty well, while others are exercises in torture.

The diving and gymnastics are similar in that you must press the joystick to various positions to control the amount of "tuck" your contestant uses. Both events end with scoring from the judges.

You dive from the ten-meter board, and you can choose a variety of dives (forward, back, reverse). You control the spinning, twisting diver on the way down. The more difficult the dive, the more possible points you can get.

The Gymnastics event has you leaping onto a springboard, then vaulting into the air (using a vaulting horse). Again, you control the gymnast as she leaps and flies through the air.

The first track event is a four-man relay. Here, you control the runner's speed, trying to go as fast as possible without using up the runner's stamina. There is also the small matter of passing the baton. The other track event is the hundred-yard dash, a "joystick buster" — just rattle the stick back and forth as fast as you can.

The two swimming events are similar. One is four-man relay, while the other is just a two-lap race for a single swimmer. You control the timing of your swimmer's start and kick-turn, and apply power to the swimmer's arms at exactly the right moment by pressing the fire button.

The best event is the skeet shoot. You control a cursor that represents the aiming point of your shotgun. A standard pattern of skeet launchers throw groups of up to four clay pigeons into the air and you must fire at them. This simple event works very well.

The most difficult and frustrating event is the pole vault. You must precisely time the placing of your pole, leaping up and over, and releasing the pole in order to make it over the bar without knocking the bar off. Unfortunately, correctly timing that final press of the fire button to let go of the pole seems virtually impossible.

No computer game can give you the actual feel of competing in a physical sport—but how many people can seriously hope to compete in the Olympics? Summer Games, with its excellent graphics, animation and sound will provide a good time overall for Olympic dreamers. ▲

Atari Corp.
1196 Borregas Avenue
Sunnyvale, CA 94086
(408) 745-2000
$24.95 or $34.95, 64K XL/XE

# SOFTWARE LIBRARY

## TYPING SPECIAL ATARI CHARACTERS

The Atari Special Characters and the keys you must type in order to get them are shown below:

For [CONTROL] key combination, *hold down* [CONTROL] while pressing the next key. For inverse [CONTROL] [A] through [CONTROL] [Z], press the [◪] key—or [⅄] on the 400/800—then *release* it before pressing the next key. (Press [◪] or [⅄] again to turn off inverse.) For [ESC] key combinations, press [ESC] and then *release* it before pressing the next key.

Carefully study the chart above and pay close attention to differences between lookalike characters such as the slash key's [/] and the [CONTROL] [F] symbol [◪].

### NORMAL VIDEO

| FOR THIS | TYPE THIS | FOR THIS | TYPE THIS |
|---|---|---|---|
| ♥ | CTRL , | | CTRL S |
| | CTRL A | | CTRL T |
| | CTRL B | | CTRL U |
| | CTRL C | | CTRL V |
| | CTRL D | | CTRL W |
| | CTRL E | | CTRL X |
| | CTRL F | | CTRL Y |
| | CTRL G | | CTRL Z |
| | CTRL H | | ESC ESC |
| | CTRL I | | ESC CTRL − |
| | CTRL J | | ESC CTRL = |
| | CTRL K | | ESC CTRL + |
| | CTRL L | | ESC CTRL * |
| | CTRL M | | CTRL . |
| | CTRL N | | CTRL ; |
| | CTRL O | | SHIFT = |
| | CTRL P | | ESC SHIFT CLEAR |
| | CTRL Q | | ESC DELETE |
| | CTRL R | | ESC TAB |

### INVERSE VIDEO

| FOR THIS | TYPE THIS |
|---|---|
| | ESC SHIFT DELETE |
| | ESC SHIFT INSERT |
| | ESC CTRL TAB |
| | ESC SHIFT TAB |
| | ⅄ CTRL . |
| | ⅄ CTRL ; |
| | ⅄ SHIFT = |
| | ESC CTRL 2 |
| | ESC CTRL DELETE |
| | ESC CTRL INSERT |

## TYPO II AUTOMATIC PROOFREADER

TYPO II automatically proofreads **Antic**'s type-in BASIC listings. Type in the listing below and SAVE a copy to disk or cassette. Now type GOTO 32000. At the prompt, type in a single program line **without the two-letter TYPO II code at the beginning**. Then press [RETURN].

Your line will reappear at the bottom of the screen. If the TYPO II code does not match the code in the magazine, then you've mistyped your line.

To call back a previously typed line, type [*], then the line number, then [RETURN]. When the completed line appears, press [RETURN] again. This is how TYPO II proofreads itself.

To LIST your program, press [BREAK] and type LIST. To return to TYPO II, type GOTO 32000. To remove TYPO II from your program, type LIST "D:FILENAME",0,31999, then [RETURN], then NEW, then ENTER "D:FILENAME", then [RETURN]. Now you can SAVE or LIST your program to disk or cassette.

Don't type the TYPO II Codes!

```
WB 32000 REM TYPO II BY ANDY BARTON
VM 32010 REM VER. 1.0 FOR ANTIC MAGAZINE
HS 32020 CLR :DIM LINE$(120):CLOSE #2:CLO
   SE #3
BN 32030 OPEN #2,4,0,"E":OPEN #3,5,0,"E"
YC 32040 ? "K":POSITION 11,1:? "TYPO II"

EM 32050 TRAP 32040:POSITION 2,3:? "Type
   in a program line"
HS 32060 POSITION 1,4:? " ":INPUT #2;LINE
   $:IF LINE$="" THEN POSITION 2,4:LIST B
   :GOTO 32060
XH 32070 IF LINE$(1,1)="*" THEN B=VAL(LIN
   E$(2,LEN(LINE$))):POSITION 2,4:LIST B:
   GOTO 32060
TH 32080 POSITION 2,10:? "CONT"
MF 32090 B=VAL(LINE$):POSITION 1,3:? " ";
NY 32100 POKE 842,13:STOP
CN 32110 POKE 842,12
ET 32120 ? "K":POSITION 11,1:? " TYPO II 
   ":POSITION 2,15:LIST B
CE 32130 C=0:ANS=C
QR 32140 POSITION 2,16:INPUT #3;LINE$:IF
   LINE$="" THEN ? "LINE ";B;" DELETED":G
   OTO 32050
VV 32150 FOR D=1 TO LEN(LINE$):C=C+1:ANS=
   ANS+(C*ASC(LINE$(D,D))):NEXT D
WJ 32160 CODE=INT(ANS/676)
JW 32170 CODE=ANS-(CODE*676)
EH 32180 HCODE=INT(CODE/26)
BH 32190 LCODE=CODE-(HCODE*26)+65
HB 32200 HCODE=HCODE+65
IE 32210 POSITION 0,16:? CHR$(HCODE);CHR$
   (LCODE)
VG 32220 POSITION 2,13:? "If CODE does no
   t match press  RETURN  and edit line a
   bove.":GOTO 32050
```

# CUSTOMER LIST MANAGER

*Article on page 8*

## LISTING 1

Don't type the TYPO II Codes!

```
FQ 10 REM CUSTOMER LIST MANAGER
KT 20 REM BY THOMAS ANDREWS
PB 30 REM (C)1989, ANTIC PUBLISHING INC.
GF 40 WON=1:ZERO=0:TEN=10:SIX=6:SIXTY=TEN*SIX:HUN=TEN*TEN:TWO=WON+WON:FIVE=5:MAX=19000
LK 45 REM THE FOLLOWING ML ROUTINE WAS WRITTEN BY SCOTT SHECK IN THE DEC 1984 ANTIC
GU 50 FIND=ADR("hh[illegible]")
BM 55 REM
OJ 60 SRCH=ADR("hh[illegible]")
RZ 65 CHNGR=ADR("hh[illegible]")
QU 70 DIM ANS$(TEN),LI(HUN),WI$(HUN),W$(SIXTY*TEN),FLD$(SIXTY),TITLE$(14),FNAME$(SIXTY),HDG$(40)
UM 80 DIM L$(MAX),SHIFT$(135)
LS 85 SHIFT$(WON)="hh[illegible]"
HK 86 SHIFT$(76)="[illegible]"
OP 90 FNAME$="NAME :PHONE:ADDR.:TYPE :NOTES:":OPEN #TWO,4,ZERO,"K:":OPEN #3,13,ZERO,"E:"
BT 100 PRINT "[illegible]":POKE 710,198:POKE 709,ZERO:POSITION SIX,ZERO:PRINT " CUSTOMER LIST MANAGER ":TRAP 25000
TK 110 PRINT "A. START NEW LIST":PRINT "B. ADD TO LIST":PFLG=ZERO:DEV=ZERO:CLOSE #WON
AP 120 PRINT "C. LOAD LIST FROM DISK":PRINT "D. EDIT/UPDATE LIST"
KN 130 PRINT "E. SAVE LIST TO DISK":PRINT "F. SORT LIST":PRINT "G. PRINT/DISPLAY LIST"
AS 140 PRINT "H. CHANGE LIST TITLE":PRINT "I. LOAD EDIT OR WORD PROCESSOR FILE"
II 180 GOSUB 20000:IF NR=ZERO THEN IF NOT ((ANS=WON) OR (ANS=3) OR (ANS=9)) THEN 100
MA 185 ON ANS GOSUB 200,1000,2000,3000,4000,5000,6000,8000,9000
LY 190 GOTO 100
JS 200 HDG$="CREATE NEW LIST":GOSUB 14000:POSITION TWO,SIX:GOSUB 20020
SY 210 L$="":NR=ZERO:LL=WON:HDG$(17)=" ITEM #":GOTO 1005
RM 1000 HDG$="ADD TO PRESENT LIST ":GOSUB 14000:HDG$(21)=" ITEM #"
ZE 1005 IF NR=HUN THEN PRINT HDG$:POSITION TWO,TEN:PRINT "THIS LIST IS FULL!":GOSUB 23000:RETURN
OH 1006 OVER=MAX-LL:IF OVER<SIX*HUN THEN PRINT HDG$:POSITION TWO,TEN:PRINT "MEMORY SPACE RUNNING LOW":GOSUB 23000
SZ 1007 ? HDG$;NR+WON:POKE 82,SIX:? :W$="":LNE=PEEK(84):POKE 702,ZERO
WU 1010 FOR J=WON TO TEN:POKE 85,ZERO:GOSUB 16000:POKE 764,255:INPUT #16,FLD$
HQ 1015 IF FLD$="" THEN FLD$=" "
XD 1020 POKE 84,LNE:PRINT "[illegible]";FLD$:X=USR(CHNGR,ADR(FLD$),LEN(FLD$))
CJ 1030 W$(LEN(W$)+WON)=FLD$:LNE=LNE+TWO:POKE 84,LNE:NEXT J:POKE 82,TWO:POKE 702,64
CG 1033 IF LEN(W$)>OVER THEN PRINT HDG$:POSITION TWO,TEN:PRINT "THIS ITEM DOES NOT FIT!":GOSUB 23000:RETURN
BW 1035 L$(LL)=W$:NR=NR+WON:LI(NR)=LL:POSITION TWO,22
JK 1040 LL=LEN(L$)+WON:? "PRESS OPTION FOR NEXT ITEM; START FOR MAIN MENU";
QE 1050 ANS=PEEK(53279):IF ANS=3 THEN 1005
OH 1060 IF ANS<>6 THEN 1050
IX 1070 L$(LL)="[illegible]":RETURN
JB 2000 HDG$="LOAD LIST FROM DISK":PRINT HDG$:GOSUB 21000
RR 2010 OPEN #WON,4,ZERO,W$:INPUT #WON;TITLE$:? :? TITLE$;"    ";
DR 2012 INPUT #WON,FLD$:YEAR=VAL(FLD$):POKE 752,WON:PRINT YEAR
WG 2015 L$="":LL=WON:NR=ZERO:TRAP 2040:GOSUB 20040
FL 2020 W$="":FOR J=WON TO TEN:INPUT #WON,FLD$:W$(LEN(W$)+WON)=FLD$:NEXT J
MJ 2025 IF LL+LEN(W$)>MAX THEN 2040
MB 2030 L$(LL)=W$:NR=NR+WON:LI(NR)=LL:LL=LEN(L$)+WON:IF NR<HUN THEN 2020
DZ 2040 POKE 752,ZERO:TRAP 25000:L$(LL)="[illegible]":CLOSE #WON:PRINT "THERE ARE ";NR;" ITEMS ON THIS LIST"
DG 2050 GOSUB 23000:RETURN
FK 3000 HDG$="EDIT/UPDATE LIST":PRINT HDG$:POSITION TWO,SIX:PRINT "WHICH FUNCTION?"
RQ 3010 PRINT "A. EDIT AND UPDATE 1 YEAR":PRINT "B. EDIT ONLY"
CT 3015 PRINT "C. UPDATE ONLY":PRINT "D. RETURN TO MAIN MENU"
ZI 3020 GOSUB 20000:MODE=ANS:IF MODE=WON THEN GOSUB 15040:YEAR=YEAR+WON:GOSUB 20040
AA 3025 IF MODE=3 THEN GOSUB 21200:YEAR=YEAR+WON:GOSUB 20040
OC 3030 ON MODE GOTO 3050,3040,3050,20180:GOTO 3000
LU 3040 PRINT HDG$:GOSUB 15000:TRAP 25000
PU 3050 OPEN #WON,8,ZERO,"D:EDITLIST.DAT":? #WON;TITLE$:? #WON;YEAR
ST 3051 I2=ZERO:FOR I=WON TO NR:IF WI$(I,I)="[illegible]" THEN 3240
TW 3053 I2=I2+WON:POKE 82,SIX:GOSUB 20100
ZB 3055 POKE 764,255:POSITION ZERO,22:POKE 82,ZERO:? "CTRL-A=NEXT ITEM   CTRL-D=DELETE ITEM"
XK 3060 PRINT "CTRL-M=MAIN MENU   ESC=REDO THIS ITEM";:POSITION SIX,TWO:PRINT "[illegible]";:POKE 82,SIX:POKE 702,ZERO
GW 3070 GET #TWO,A:IF A=125 OR A=156 OR A=157 OR A=254 OR A=255 THEN 3070
NU 3075 IF A=155 THEN POKE 85,SIX:PUT #3,29:GOTO 3070
BB 3080 IF A=WON OR A=13 OR A=4 THEN 3100
FG 3085 IF A=27 THEN I=I-WON:I2=I2-WON:NEXT I:RETURN
OS 3090 PUT #3,A:GOTO 3070
TR 3100 IF A<>4 THEN 3200
UH 3105 SOUND ZERO,SIXTY,TEN,TEN:FOR DELAY=WON TO SIXTY:NEXT DELAY:SOUND ZERO,ZERO,ZERO,ZERO
FE 3110 POKE 82,ZERO:POKE 712,52:POSITION 2,22:PRINT "[illegible]ARE YOU SURE? (Y/N)";:INPUT ANS$
XL 3120 POKE 712,ZERO:IF ANS$="" THEN 3126
XB 3125 IF ANS$(WON,WON)="Y" OR ANS$(WON,WON)="y" THEN 3230
TJ 3126 I=I-WON:I2=I2-WON:NEXT I:RETURN
```

```
YF 3200 LNE=TWO:POKE 82,SIX:POKE 85,SIX:F
   OR J=WON TO TEN:POKE 84,LNE:INPUT #3,F
   LD$:IF FLD$="" THEN FLD$=" "
MM 3210 X=USR(CHNGR,ADR(FLD$),LEN(FLD$)):
   PRINT #WON;FLD$:LNE=LNE+TWO:NEXT J
FO 3230 POKE 702,64:POKE 82,TWO:GOTO 3255
IM 3240 POKE 752,WON:PRINT HDG$:POSITION
   TEN,TEN:PRINT "PROCESSING ITEM ";I
RC 3245 RX1=LI(I)-WON:LAD1=RX1+ADR(L$):FO
   R J=WON TO TEN:BEGIN1=USR(FIND,LAD1,J)
   +RX1:LAST1=PEEK(WON)+BEGIN1-WON
AH 3246 IF J=SIX THEN IF MODE=3 THEN NEXT
   J
SJ 3250 PRINT #WON;L$(BEGIN1,LAST1):NEXT
   J:IF MODE=3 THEN PRINT #WON;"■"
SP 3253 GOTO 3260
SJ 3255 IF MODE=WON AND A=13 AND I2<ITEMS
    THEN 3380
IY 3256 IF A=13 AND I<>NR THEN GOSUB 2120
   0
OQ 3260 NEXT I:POKE 752,ZERO:CLOSE #WON
SP 3270 TRAP 25000:W$="D:EDITLIST.DAT":PR
   INT HDG$:GOTO 2010
UT 3380 PRINT "■":POSITION TWO,TEN:PRINT
   "THE ENTIRE LIST MUST BE REVIEWED WHEN
    UPDATING"
AS 3390 GOSUB 23000:GOTO 3260
ZI 4000 HDG$="■SAVE LIST TO DISK":PRINT H
   DG$:GOSUB 15000:PRINT HDG$:GOSUB 21000
   :TRAP 25000
ER 4010 OPEN #WON,8,ZERO,W$:? #WON;TITLE$
   :? #WON;STR$(YEAR)
BF 4020 FOR I=WON TO NR:IF WI$(I,I)="■" T
   HEN 4050
QC 4030 RX1=LI(I)-WON:LAD1=RX1+ADR(L$):FO
   R J=WON TO TEN:BEGIN1=USR(FIND,LAD1,J)
   +RX1:LAST1=PEEK(WON)+BEGIN1-WON
HR 4040 PRINT #WON;L$(BEGIN1,LAST1):NEXT
   J
TQ 4050 NEXT I:CLOSE #WON:RETURN
VM 5000 HDG$="■SORT LIST":PRINT HDG$:PRIN
   T TITLE$;"  ";YEAR
YN 5010 I=WON:GOSUB 20150:A=ANS:IF A=TEN+
   WON THEN RETURN
DB 5020 I=TWO:GOSUB 20150:B=ANS:IF B=TEN+
   WON THEN RETURN
MR 5025 PRINT HDG$:POSITION TWO,TEN:PRINT
   "CHECKING ITEM # ":POKE 752,WON
FP 5030 LAD=ADR(L$):LIADR=ADR(ANS$)+TEN+S
   IX:FOR I=NR TO TWO STEP -WON:I1=I:I2=I
   -WON
KC 5035 POSITION 19,TEN:PRINT I;"  "
PH 5060 RX1=LI(I1)-WON:LAD1=LAD+RX1
YJ 5065 BEGIN1=USR(FIND,LAD1,A)+RX1:LAST1
   =PEEK(WON)+BEGIN1-WON
SQ 5090 RX2=LI(I2)-WON:LAD2=LAD+RX2
DE 5100 BEGIN2=USR(FIND,LAD2,A)+RX2:LAST2
   =PEEK(WON)+BEGIN2-WON
GM 5130 IF L$(BEGIN1,LAST1)>L$(BEGIN2,LAS
   T2) THEN 5500
JI 5135 IF L$(BEGIN1,LAST1)<L$(BEGIN2,LAS
   T2) THEN 5160
FZ 5140 STRT1=USR(FIND,LAD1,B)+RX1:FIN1=P
   EEK(WON)+STRT1-WON
MQ 5145 STRT2=USR(FIND,LAD2,B)+RX2:FIN2=P
   EEK(WON)+STRT2-WON
YE 5150 IF L$(STRT1,FIN1)>=L$(STRT2,FIN2)
    THEN 5500
JU 5160 FOR J=I TO NR:J1=J:IF J=NR THEN 5
   260
JA 5190 I3=J+WON:RX3=LI(I3)-WON
RU 5200 LAD3=LAD+RX3:BEGIN3=USR(FIND,LAD3
   ,A)+RX3:LAST3=PEEK(WON)+BEGIN3-WON
FG 5230 IF L$(BEGIN2,LAST2)<L$(BEGIN3,LAS
   T3) THEN J=NR
IH 5235 IF L$(BEGIN2,LAST2)<>L$(BEGIN3,LA
   ST3) THEN 5260
RV 5240 STRT3=USR(FIND,LAD3,B)+RX3:FIN3=P
   EEK(WON)+STRT3-WON
MS 5245 STRT2=USR(FIND,LAD2,B)+RX2:FIN2=P
   EEK(WON)+STRT2-WON
MB 5250 IF L$(STRT2,FIN2)<=L$(STRT3,FIN3)
    THEN J=NR
GA 5260 NEXT J
IU 5290 K=(J1-I2)*SIX:X=USR(ADR(SHIFT$),L
   IADR+(I1-WON)*SIX,LIADR+(I2-WON)*SIX,K
   ):LI(J1)=RX2+WON
PT 5500 NEXT I:POKE 752,ZERO:RETURN
YI 6000 HDG$="■PRINT/DISPLAY LIST"
LT 6005 PRINT HDG$:POSITION TWO,SIX:PRINT
   "DISPLAY DEVICE?■■"
IH 6010 PRINT "A. SCREEN■":PRINT "B. PRIN
   TER■":? "C. WORD PROCESSOR FILE■":PRIN
   T "D. RETURN TO MAIN MENU"
NL 6020 GOSUB 20000:ON ANS GOTO 6030,6100
   ,6400,21020:GOTO 6005
QP 6030 MODE=TWO:HDG$="■DISPLAY LIST":GOS
   UB 15000:I2=ZERO
MK 6035 FOR I=WON TO NR:IF WI$(I,I)="■" T
   HEN 6090
AR 6037 POKE 82,SIX:LNE=TWO:I2=I2+WON:GOS
   UB 20100
UR 6040 POKE 82,TWO:POSITION TWO,22:PRINT
   "OPTION=FORWARD SELECT=BACK START=MEN
   U";
XG 6050 A=PEEK(53279):IF (A=3)+(A=5)+(A=S
   IX)=ZERO THEN 6050
ES 6060 IF A=3 THEN IF I2=ITEMS THEN I=I-
   WON:I2=I2-WON
ET 6065 IF A=3 THEN 6090
NW 6070 IF A=SIX THEN I=NR:GOTO 6090
WH 6080 I2=I2-TWO:IF I2<ZERO THEN I=ZERO:
   I2=ZERO:GOTO 6090
SJ 6082 FOR I1=I-WON TO WON STEP -WON:IF
   WI$(I1,I1)<>"■" THEN I=I1-WON:I1=WON:G
   OTO 6084
AI 6083 IF I1=WON THEN I=ZERO:I2=ZERO
CY 6084 NEXT I1
HD 6090 NEXT I:RETURN
IL 6100 HDG$="■PRINT LIST":GOSUB 15000:NP
   =INT((ITEMS+4)/FIVE):PRINT HDG$;"■■■■"
   :DEV=WON
ZL 6110 PRINT "THERE ARE ";NP;" PAGES TO
   THIS LIST■"
PU 6115 PRINT "WHICH PAGES ARE TO BE PRIN
   TED?■■"
MQ 6117 PRINT "(A '0' ANSWER INDICATES AL
   L PAGES)■■■"
WX 6120 TRAP 6120:PRINT "FIRST PAGE? ";:I
   NPUT #16,FP:FP=INT(FP):IF FP=ZERO THEN
    FP=WON:LP=NP:GOTO 6140
GT 6125 IF FP<WON OR FP>NP THEN 6120
CT 6130 TRAP 6130:PRINT "LAST PAGE? ";:IN
   PUT #16,LP:LP=INT(LP):IF LP=ZERO THEN
   FP=WON:LP=NP
YF 6135 IF LP<FP OR LP>NP THEN 6130
KV 6140 PRINT "■■■■■■■ADJUST PAPER, THEN P
   RESS ANY KEY":GET #TWO,ANS
TA 6150 TRAP 25000:OPEN #WON,8,ZERO,"P:":
   PFLG=WON:WP=WON:ANS$="      "
CK 6200 K=-WON:PG=ZERO:FOR I=WON TO NR:IF
    WI$(I,I)="■" THEN 6350
WB 6210 K=K+WON:X=K/FIVE:IF INT(X)=X THEN
    6220
XI 6215 IF PG>=FP AND PG<=LP THEN 6325
UC 6217 GOTO 6350
DZ 6220 PG=PG+WON:IF PG>LP THEN I=NR:GOTO
    6350
CD 6230 IF PG<FP THEN 6350
UA 6310 PRINT "■PRINTING...■■■":? "PRESS
   ANY KEY TO STOP"
CJ 6315 JJ=40-(LEN(TITLE$)+14)/TWO:W$=" "
   :W$(JJ)=W$:W$(TWO)=W$
CB 6320 PRINT #WON;W$;TITLE$;"    ";YEAR;"
    Pg.";PG
LF 6325 RX1=LI(I)-WON
CJ 6330 PRINT #WON:FOR J=WON TO TEN:IF PE
   EK(764)<255 THEN I=NR:J=TEN:POKE 764,2
   55:GOTO 6345
WE 6340 GOSUB 20400:PRINT #WON;ANS$;:GOSU
   B 16000:PRINT #WON;" ";FLD$
DU 6345 NEXT J:IF WP THEN IF INT((K+WON)/
   FIVE)=(K+WON)/FIVE THEN FOR J=WON TO T
   EN:PRINT #WON:NEXT J
TY 6350 NEXT I:CLOSE #WON:RETURN
ES 6400 HDG$="■PRINT WORD PROCESSOR FILE"
   :GOSUB 15000
DR 6410 PRINT HDG$:GOSUB 21000:OPEN #WON,
   8,ZERO,W$:WP=ZERO:PFLG=WON:ANS$="":DEV
   =WON
HH 6420 PRINT #WON;TITLE$;"    ";YEAR
KC 6430 FOR I=WON TO NR:IF WI$(I,I)="■" T
   HEN 6350
US 6440 GOTO 6325
MA 8000 HDG$="■CHANGE LIST TITLE":GOSUB 1
   4000:POSITION TWO,5:PRINT "PRESENT LIS
   T TITLE: ";TITLE$
KY 8010 PRINT :PRINT "YEAR: ";YEAR;"■■■":
   GOSUB 20020:RETURN
```

*continued on next page*

```
TG 9000 HDG$="▣RECOVER LIST":PRINT HDG$:P
   OSITION TWO,SIX
XD 9010 PRINT "WHICH FILE TYPE?⊞⊞":PRINT
   "A. EDIT FILE⊞":PRINT "B. WORD PROCESS
   OR FILE⊞"
OS 9020 PRINT "C. RETURN TO MAIN MENU":GO
   SUB 20000:ON ANS GOTO 9040,9050,20015:
   GOTO 9000
RF 9040 W$="D:EDITLIST.DAT":PRINT HDG$:GO
   TO 2010
NE 9050 PRINT HDG$:GOSUB 21000:OPEN #WON,
   4,ZERO,W$:NR=ZERO:L$="":LL=WON:TRAP 91
   00
MN 9060 INPUT #1,FLD$:X=LEN(FLD$):YEAR=VA
   L(FLD$(X-3,X)):TITLE$=FLD$(WON,X-7):PR
   INT FLD$
EA 9070 INPUT #WON,FLD$:FOR J=WON TO TEN:
   INPUT #1,W$:FLD$=W$(8):X=USR(CHNGR,ADR
   (FLD$),LEN(FLD$))
JY 9075 IF LEN(L$)+LEN(FLD$)+WON>MAX THEN
    9100
XK 9080 L$(LEN(L$)+WON)=FLD$:NEXT J:NR=NR
   +WON:LI(NR)=LL:LL=LEN(L$)+WON:IF NR<HU
   N THEN 9070
FU 9100 L$(LL)="■":CLOSE #WON:RETURN
YC 14000 POKE 709,TEN:POKE 710,ZERO:PRINT
    HDG$:POSITION TEN,TEN
OX 14010 PRINT "BAILOUT OPTION!!!⊞":? "IF
    YOU BELONG HERE,PRESS RETURN"
QD 14020 PRINT "FOR THE MAIN MENU, PRESS
   THE SPACE BAR"
FO 14030 GET #2,ANS:IF ANS=32 THEN POP :R
   ETURN
OA 14040 IF ANS<>155 THEN PRINT CHR$(ANS)
   ;:GOTO 14030
UR 14050 POKE 709,ZERO:POKE 710,198:PRINT
    HDG$:RETURN
OB 15000 PRINT HDG$:POSITION TWO,SIX:PRIN
   T "HOW MANY ITEMS ARE TO BE SELECTED?⊞
   ⊞"
SS 15010 PRINT "A. ALL ITEMS⊞":PRINT "B.
   SOME ITEMS⊞":PRINT "C. RETURN TO MAIN
   MENU":GOSUB 20000
GH 15020 ON ANS GOTO 15040,15050,15030:GO
   TO 15000
YL 15030 POP :RETURN
CI 15040 ITEMS=NR:WI$="⊞":WI$(NR)=WI$:WI$
   (TWO)=WI$:RETURN
IM 15050 PRINT HDG$:I=WON:GOSUB 20150:A=A
   NS:IF A=TEN+WON THEN RETURN
RB 15055 LAD=ADR(L$):PRINT HDG$:POKE 702,
   64
FI 15060 POSITION TWO,SIX:PRINT "SEARCH S
   EQUENCE?":? "(USE '?' FOR A WILDCARD C
   HARACTER)"
TP 15065 PRINT "(USE ALL CAPITAL LETTERS)
   ⊞":INPUT FLD$:IF FLD$="" THEN FLD$=" "

IA 15070 PRINT HDG$:POSITION 12,8:PRINT "
   SEARCHING...":POSITION TEN,TEN:PRINT "
   ITEMS FOUND:"
SV 15100 GOSUB 21200:ITEMS=ZERO
HV 15110 POKE 752,WON:FOR I=WON TO NR:RX1
   =LI(I)-WON:POSITION 23,8:PRINT I
NQ 15120 LAD1=LAD+RX1:BEGIN1=USR(FIND,LAD
   1,A)+RX1:LAST1=PEEK(WON)+BEGIN1-WON:W$
   =L$(BEGIN1,LAST1)
CB 15125 W$(WON,WON)=CHR$(ASC(W$(WON,WON)
   )-128)
QW 15130 X=USR(SRCH,ADR(FLD$),LEN(FLD$),A
   DR(W$),LEN(W$))
TZ 15140 IF X THEN WI$(I,I)="⊞":ITEMS=ITE
   MS+WON
RW 15150 POSITION 23,TEN:PRINT ITEMS:NEXT
    I:POKE 752,ZERO:IF ITEMS<>ZERO THEN R
   ETURN
VY 15160 PRINT HDG$:PRINT "⊞⊞⊞⊞NO ITEMS F
   OUND!!!⊞⊞":PRINT "FIELD #";A;"⊞⊞"

QL 15170 PRINT "SEARCH SEQUENCE:":PRINT F
   LD$:POSITION TWO,20:PRINT "PRESS RETUR
   N"
CK 15180 INPUT ANS$:POP :RETURN
XH 16000 PRINT #DEV;FNAME$(J*SIX-FIVE,J*S
   IX);:RETURN
BD 20000 SOUND ZERO,125,TEN,TEN:FOR DELAY
   =WON TO TEN:NEXT DELAY:SOUND ZERO,ZERO
   ,ZERO,ZERO:POKE 764,255
BF 20005 POSITION TWO,21:PRINT "ENTER LET
   TER: ";:INPUT #16,ANS$:IF ANS$="" THEN
    20000
NH 20010 ANS=ASC(ANS$(WON,WON))-64:IF ANS
   <ZERO THEN ANS=ZERO
EB 20015 RETURN
YI 20020 PRINT "LIST TITLE? (14 CHAR. MAX
   ) ":INPUT TITLE$:PRINT
CI 20030 TRAP 20030:PRINT "YEAR ":INPUT Y
   EAR:TRAP 25000
PJ 20035 IF YEAR<1000 OR YEAR>2100 THEN 2
   0030
NH 20040 FNAME$=FNAME$(WON,30)
KJ 20045 FOR I=YEAR-FIVE TO YEAR-WON:FNAM
   E$(LEN(FNAME$)+WON)=STR$(I):FNAME$(LEN
   (FNAME$)+WON)=" :":NEXT I:RETURN
PG 20100 POKE 752,WON:PRINT "▣":POSITION
   TWO,ZERO:? TITLE$;" ";YEAR;"  ITEM ";I
   2;" OF ";ITEMS:? :RX1=LI(I)-WON
YC 20110 FOR J=WON TO TEN:POKE 85,ZERO:GO
   SUB 16000
FT 20115 JJ=J:IF MODE=WON AND J=TEN THEN
   20130
XF 20116 IF MODE=WON AND J>=SIX THEN JJ=J
   +WON
GC 20120 GOSUB 20400:PRINT FLD$
UY 20130 NEXT J:POKE 752,ZERO:RETURN
XP 20150 POSITION TWO,SIX:PRINT "WHICH FI
   ELD IS TO BE USED FOR LEVEL ";I;"?⊞"
SG 20160 FOR J=WON TO TEN:PRINT CHR$(64+J
   );". ";FNAME$(J*SIX-FIVE,J*SIX-WON):NE
   XT J
IS 20165 PRINT "K. RETURN TO MAIN MENU"
EJ 20170 GOSUB 20000:POSITION TWO,21:PRIN
   T "▣":IF ANS<WON OR ANS>TEN+WON THEN 2
   0170
PP 20175 IF ANS=TEN+WON THEN POP :RETURN
EH 20180 RETURN
MZ 20400 LAD1=ADR(L$)+RX1:BEGIN1=USR(FIND
   ,LAD1,J)+RX1:LAST1=PEEK(WON)+BEGIN1-WO
   N
TD 20410 FLD$=L$(BEGIN1,LAST1):FLD$(WON,W
   ON)=CHR$(ASC(FLD$(WON,WON))-128):IF PF
   LG THEN RETURN
SM 20420 B=LEN(FLD$)+WON:IF B>40 THEN RET
   URN
OJ 20430 FLD$(B)=" ":FLD$(SIXTY)=" ":FLD$
   (B+WON)=FLD$(B):RETURN
EH 21000 POSITION 2,6:PRINT "FILENAME?":P
   RINT "BLANK ENTRY RETURNS TO MAIN MENU
   ⊞":INPUT W$
NI 21005 IF W$="" THEN POP :RETURN
LE 21007 FLD$="D:":IF LEN(W$)>WON THEN IF
    W$(WON,TWO)="D:" THEN RETURN
QU 21010 FLD$(3)=W$:W$=FLD$
DI 21020 RETURN
KY 21200 WI$="▣":WI$(HUN)=WI$:WI$(TWO)=WI
   $:RETURN
UT 23000 FOR DELAY=WON TO SIXTY*FIVE:NEXT
    DELAY:RETURN
XH 25000 PRINT HDG$;"⊞⊞⊞⊞⊞":X=PEEK(195):P
   RINT " ERROR #";X:POSITION TWO,TEN+TEN
   :PRINT "PRESS RETURN":CLOSE #WON
QY 25010 POKE 752,ZERO:POKE 82,TWO:TRAP 2
   5000:INPUT #16;W$:POP :GOTO 100
```

## LISTING 2

```
AY 10 REM
AZ 20 REM
GD 30 REM (c) 1985,1988 ANTIC PUBLISHING
EV 40 REM (LINES 10-250 MAY BE USED WITH
   OTHER BASIC LOADERS IN THIS ISSUE.
IJ 50 REM CHANGE LINE 70 AS NECESSARY.)
PR 60 DIM FN$(20),TEMP$(20),AR$(93):DPL=P
   EEK(10592):POKE 10592,255
WO 70 FN$="D:LINES.LST":REM THIS IS THE N
   AME OF THE DISK FILE TO BE CREATED
RD 80 ? "▣Disk or Cassette?";:POKE 764,25
   5
PY 90 IF  NOT (PEEK(764)=18 OR PEEK(764)=
   58) THEN 90
TH 100 IF PEEK(764)=18 THEN FN$="C:"
VB 110 POKE 764,255:GRAPHICS 0:? "      AN
   TIC'S GENERIC BASIC LOADER"
MY 120 ? ,"BY CHARLES JACKSON"
KB 130 POKE 10592,DPL:TRAP 200
PU 140 ? :? :? "Creating ";FN$:? "...plea
   se stand by."
LW 150 RESTORE :READ LN:LM=LN:DIM A$(LN):
   C=1
BQ 160 AR$="":READ AR$
YC 170 FOR X=1 TO LEN(AR$) STEP 3:POKE 75
```

```
    2,255
DM  180 LM=LM-1:POSITION 10,10:? "(Countdo
    wn...T-";INT(LM/10);")      "
BK  190 A$(C,C)=CHR$(VAL(AR$(X,X+2))):C=C+
    1:NEXT X:GOTO 160
MM  200 IF PEEK(195)=5 THEN ? :? :? "TOO
    MANY DATA LINES!":? "CANNOT CREATE FIL
    E!":END
CM  210 IF C<LN+1 THEN ? :? "TOO FEW DATA
     LINES!":? "CANNOT CREATE FILE!":END
UQ  220 IF FN$="C:" THEN ? :? " Prepare ca
    ssette, press [RETURN]"
AR  230 OPEN #1,8,0,FN$
PV  240 POKE 766,1:? #1;A$;:POKE 766,0
AL  250 CLOSE #1:GRAPHICS 0:? " COMPLETED 
    "
GC  1000 DATA 421
PJ  1010 DATA 0530480320700730780680610650
    6808204003410410413300110413300010413З
    003104133002169001133212160
LM  1020 DATA 0001322131770000160181980022
    08014165003208008200177000016251132001
    096198003230212208002230213
RF  1030 DATA 2300002082242300012082200340
    4115505404803208308206707206106506808２
    040034104104133204104133203
JU  1040 DATA 1041041332051041332071041332
    0610413320910413320816900013321213321З
    165205240063165209208006165
PF  1050 DATA 2081972051440531600001772032
    0106324003804112720920624003220106514４
    011201091176007024105032209
NK  1060 DATA 2062400171652082080021982091
    982082302062080022302070241442002001９6
    205208207230212096034041155
OA  1070 DATA 0540530320670720780710820610
    6506808204003410410413320410413320310４
    133206104133205160000177203
UC  1080 DATA 0091281452031652052080021982
    0619820520800516520620800109623020320８
    002230204169127049203056176
DS  1090 DATA 2280340411550560530320830720
    730700840360400870790780410610341041０4
    133204104133203104133208104
UU  1100 DATA 1332071041332131332061041332
    1213320520800219820619820516000016200１
    056165203229207133209165204
HH  1110 DATA 2292080052092400851760272320
    2416520310120513320316520410120613320４
    024165207101205133207165208
RX  1120 DATA 1012060341550560540320830720
    7307008403604005505404106103413320817７
    203145207165212208002198213
YM  1130 DATA 1982122080041652132400382240
    0224001423020320800223020423020720822４
    230208208220165203208002198
TZ  1140 DATA 2041982031652072080021982081
    982072082022402000960341５5
```

FASTER FRACTALS GROW IN YOUR ATARI

# SUPERHOP ACTION!

*Article on page 20*

## *LISTING 1*

```
;SUPERHOP ACTION!
;BY DOUGLAS SKRECKY
;(C)1989, ANTIC PUBLISHING INC.
;
DEFINE BEGIN="DO",ENDWHILE="OD",
       ENDIF="FI",END="OD"
;
MODULE
 CARD ARRAY LINE(192)
 BYTE ARRAY RSH3(320),FILENAME(16)
 BYTE KEY=764,DISPLAY=559,IOTYPE=850,
      SCREEN_SIZE,R1,R2,Z,MENUKEY,
      CLEAR,LARGER,SMALLER,UP,DOWN,
      LEFT,RIGHT,SPACEBAR,LOAD,SAVE,
      CREATE,DEMO,DEMOMODE,ON,OFF,PY
 INT  CX,CY,X,Y,XX,I,J,IC,JC,X0,Y0
 CARD BUFFER_ADDRESS=852,COUNT,PX,
      BUFFER_LENGTH=856,SCREEN=88
;
PROC INITIALIZE_PLOT()
 GRAPHICS(8+16) SETCOLOR(2,0,0)
 FOR PY=0 TO 191 BEGIN
   LINE(PY)=SCREEN+40*PY
 END
 FOR PX=0 TO 319 BEGIN
   RSH3(PX)=PX RSH 3
 END
RETURN
;
PROC PLOT(CARD PX,BYTE PY)
 BYTE POINTER PT
 BYTE ARRAY ON=[128 64 32 16 8 4 2 1]
 PT=LINE(PY) + RSH3(PX)
 PT^==%ON(PX&7)
RETURN
;
PROC CIO=$E456(BYTE AREG,XREG) []
;
PROC DISKDIR()
 BYTE ARRAY FILES(20)
 GRAPHICS(0)
 SETCOLOR(2,9,0) SETCOLOR(4,9,0)
 CLOSE(1) OPEN(1,"D1:*.*",6,0)
 PRINTE("              DISK DIRECTORY              ")
 INPUTSD(1,FILES)
 WHILE FILES(16)<>'S BEGIN
   PRINTF("%S%S",FILES,"  ")
   INPUTSD(1,FILES)
 ENDWHILE
 PRINTF("%E%S%S%E","                ",FILES)
 CLOSE(1)
RETURN
;
PROC CHOOSE_FILE()
 BYTE ARRAY TEMP(16)
 PRINT("==>")
 ZERO(FILENAME,16)
 FILENAME(1)='D
 FILENAME(2)='1
 FILENAME(3)=':
 KEY=CLEAR INPUTS(TEMP)
 SASSIGN(FILENAME,TEMP,4,16)
RETURN
;
PROC LOAD_FRACTAL()
 DISKDIR()
 PRINTE("  TYPE IN FILENAME TO LOAD PICTURE  ")
 CHOOSE_FILE()
 CLOSE(1)
 OPEN(1,FILENAME,4,0)
 GRAPHICS(8+16) SETCOLOR(2,0,0)
 IOTYPE=7
 BUFFER_ADDRESS=SCREEN
 BUFFER_LENGTH=7680
 CIO(0,16)
 CLOSE(1)
 KEY=CLEAR
 DO UNTIL KEY=LOAD OR KEY=DEMO OR
  KEY=CREATE OR KEY=MENUKEY
 END
 INITIALIZE_PLOT()
RETURN
;
PROC SAVE_FRACTAL()
 BYTE ARRAY SAVESCREEN(7680)
 MOVEBLOCK(SAVESCREEN,SCREEN,7680)
 DISKDIR()
 PRINTE("  TYPE IN FILENAME TO SAVE PICTURE  ")
 CHOOSE_FILE()
 CLOSE(1)
 OPEN(1,FILENAME,8,0)
```

*continued on next page*

```
 IOTYPE=11
 BUFFER_ADDRESS=SAVESCREEN
 BUFFER_LENGTH=7680
 CIO(0,16)
 CLOSE(1)
 INITIALIZE_PLOT()
 MOVEBLOCK(SCREEN,SAVESCREEN,7680)
RETURN
;
PROC MENU()
 GRAPHICS(0) POKE(752,1)
 SETCOLOR(2,9,0) SETCOLOR(4,9,0)
 PRINT("████████████████████SUPERHOP████████████████████")
 PUTE() PUTE() PUTE() PUTE()
 PRINTE(" Press M To Return To THIS Menu")
 PRINTE(" Press D To Turn Demomode On")
 PRINTE(" Press C To Create a Fractal")
 PRINTE(" Press L To Load a Fractal Picture")
 PRINTE(" Press S To Save a Fractal Picture")
 PRINTE(" Press < To Zoom-out For a Wider View")
 PRINTE(" Press > To Zoom-in Or Magnify")
 PRINTE(" Use Cursor Keys To Move Fractal")
 PRINTE(" Press Spacebar To Toggle Display")
 PUTE() PUTE() PUTE() PUTE()
 PRINTE("          BY Douglas Skrecky")
 KEY=CLEAR
 WHILE KEY=CLEAR OR KEY=SAVE OR KEY=SPACEBAR
 BEGIN ENDWHILE
 INITIALIZE_PLOT()
RETURN
;
PROC SETUP()
 CLEAR=255 CREATE=18 DEMO=58
 MENUKEY=37 LARGER=55 SMALLER=54
 UP=14 DOWN=15 LEFT=6 RIGHT=7
 LOAD=0 SAVE=62 SPACEBAR=33
 ON=34 OFF=0 DEMOMODE=OFF
RETURN

PROC NEW_PARAMETERS()
 BYTE ATTRACT=77
 ATTRACT=0
 X=0 Y=0 Z=0 COUNT=0 X0=0 Y0=0
 CX=200+2*RAND(0)+RAND(0)
 CY=200+2*RAND(0)+RAND(0)
 R1=1+RAND(5) R2=2+RAND(4)
 JC=96 IC=160 SCREEN_SIZE=6
RETURN
;
PROC DRAW_FRACTAL()
 BYTE ATTRACT=77
 ATTRACT=0
 IF Y>=0 THEN J=JC+ (Y RSH SCREEN_SIZE)
  ELSE J=JC- (-Y RSH SCREEN_SIZE)
 ENDIF
 IF X>=0 THEN XX=Y+((X RSH R1)+CX)
   IF J>0 AND J<191 THEN I=IC+(X RSH SCREEN_SIZE)
    IF I>0 AND I<319 THEN PLOT(I,J)
    ENDIF
   ENDIF
  ELSE XX=Y+((-X RSH R2)-CX)
   IF J>0 AND J<191 THEN I=IC-(-X RSH SCREEN_SIZE)
    IF I>0 AND I<319 THEN PLOT(I,J)
    ENDIF
   ENDIF
 ENDIF
 Y=(CY-X) Z==+1 X=XX
 IF Z=0 THEN Y==-7
 ENDIF
 IF DEMOMODE=ON THEN COUNT==+1
  IF COUNT>15000 THEN KEY=DEMO
  ENDIF
 ENDIF
RETURN
;
PROC CHANGE_PARAMETERS()
 IF KEY=MENUKEY THEN MENU() INITIALIZE_PLOT()
 ENDIF
 IF KEY=SPACEBAR THEN
  IF DISPLAY=OFF THEN DISPLAY=ON
   ELSE DISPLAY=OFF
  ENDIF
  KEY=CLEAR RETURN
 ENDIF
 IF KEY=SAVE THEN SAVE_FRACTAL() RETURN
 ENDIF
 IF KEY=LOAD THEN LOAD_FRACTAL() NEW_PARAMETERS() RETURN
 ENDIF
 IF KEY=DEMO THEN DEMOMODE=ON NEW_PARAMETERS()
  ELSE DEMOMODE=OFF
 ENDIF
 IF KEY=CREATE THEN NEW_PARAMETERS()
 ENDIF
 IF KEY=LARGER AND SCREEN_SIZE>1 THEN SCREEN_SIZE==-1
  IC=2*IC-162 JC=2*JC-96
 ENDIF
 IF KEY=SMALLER AND SCREEN_SIZE<8 THEN SCREEN_SIZE==+1
  IC=(162+IC)/2 JC=(96+JC)/2
 ENDIF
 IF KEY=LEFT THEN  IC==-10
 ENDIF
 IF KEY=RIGHT THEN IC==+10
 ENDIF
 IF KEY=UP THEN    JC==-10
 ENDIF
 IF KEY=DOWN THEN  JC==+10
 ENDIF
 ZERO(SCREEN,7680)
 X=0 Y=0 Z=0 COUNT=0
 KEY=CLEAR
RETURN
;
PROC HOPDEMO()
 SETUP()
 MENU()
 NEW_PARAMETERS()
 BEGIN
   WHILE KEY=CLEAR BEGIN
     DRAW_FRACTAL()
   ENDWHILE
 CHANGE_PARAMETERS()
 END
RETURN
```

## ANIMATION WITHOUT PLAYER/MISSILES

# HI-RES PUT AND GET

*Article on page 15*

## LISTING 1

Don't type the TYPO II Codes!

```
CD 3 REM HIGH RESOLUTION PUT AND GET
UV 4 REM REVISION 1.1
XT 5 REM BY BRAD TIMMINS
FJ 6 REM (C)1989, ANTIC PUBLISHING INC.
DW 7 REM ADD ONE TO XWIDTH AND YLENGTH
UT 8 REM TO CALCULATE STORAGE SPACE IF
PS 9 REM IMAGE IS TO BE STORED STRING
RH 10 DIM IMAGE$(182):XSTART=11:YSTART=11
   :XWIDTH=18:YLENGTH=18:HGET=0:HPUT=1:GM
   ODE=7:DX=+1:DY=+1:P=0
VU 20 GRAPHICS 7+16
MA 30 SETCOLOR 4,0,0:SETCOLOR 1,3,4:SETCO
   LOR 0,1,12
MG 40 GOSUB 1000
ZZ 50 REM ******DRAW BORDERS******
UO 60 COLOR 2:PLOT 0,0:DRAWTO 159,0:DRAWT
   O 159,95:DRAWTO 0,95:DRAWTO 0,0
FP 70 REM ******DRAW BALL******
IF 80 C=1:B=2:R=8
CL 90 FOR I=0 TO 1
GP 100 FOR X=-R TO R
KJ 110 IF C=1 THEN C=2:B=1:GOTO 130
LR 120 C=1:B=2
EJ 130 Y=SQR(R*R-X*X)
SJ 140 COLOR C:PLOT 20+X,20+Y:DRAWTO 20+X
   ,20-Y
WB 150 COLOR B:PLOT 20+X,18:DRAWTO 20+X,2
   2
LX 160 NEXT X
VM 170 REM ******SAVE IMAGES******
QP 180 HGET=USR(1536,XSTART,YSTART,XWIDTH
   ,YLENGTH,ADR(IMAGE$)+P,GMODE,HGET):P=P
   +91:C=2:NEXT I
HD 190 P=0
RX 200 REM ******MOVE BALL******
TQ 210 IF XSTART=1 OR XSTART=158-XWIDTH T
   HEN DX=-DX
RW 220 IF YSTART=1 OR YSTART=94-YLENGTH T
   HEN DY=-DY
WW 230 P=P+91:IF P=182 THEN P=0
```

```
JI  240 XSTART=XSTART+DX:YSTART=YSTART+DY
BJ  250 HPUT=USR(1536,XSTART,YSTART,XWIDTH
    ,YLENGTH,ADR(IMAGE$)+P,GMODE,HPUT)
MO  260 GOTO 210

NF  975 REM ******************************
MZ  980 REM High resolution PUT and GET
JV  985 REM machine language subroutine.
MU  990 REM ******************************
MU  995 REM X=USR(1536,XSTART,YSTART,XWIDT
    H,YLENGTH,IMAGE,GRAPHICS MODE,COMMAND)

RK  1000 FOR T=1536 TO 1773:READ A:POKE T,
    A:NEXT T
EW  1005 FOR T=1021 TO 1143:READ A:POKE T,
    A:NEXT T
BB  1007 RETURN
YH  1010 DATA 162,13,104,104,149,212,202,1
    6,250,165,214,201,3,144,4,201,14,144,1
    ,96,56
JD  1020 DATA 165,214,233,3,170,189,79,4,1
    33,231,189,90,4,133,232,189,101,4,133,
    234,165
MG  1030 DATA 88,133,203,165,89,133,204,16
    5,212,240,72,32,253,3,165,228,133,229,
    32,211,6
IO  1040 DATA 166,234,160,0,177,216,164,22
    9,37,227,240,9,177,203,5,226,145,203,2
    4,144,8
FV  1050 DATA 165,226,73,255,49,203,145,20
    3,70,226,208,3,32,197,6,70,227,208,3,3
    2,202
DY  1060 DATA 6,202,208,212,198,235,208,20
    6,198,236,240,202,32,224,6,198,218,208
    ,188,96,32
CD  1070 DATA 253,3,165,228,133,229,32,211
    ,6,166,234,164,229,177,203,160,0,37,22
    6,240,9
ES  1080 DATA 177,216,5,227,145,216,24,144
    ,8,165,227,73,255,49,216,145,216,70,22
    6,208,3
FC  1090 DATA 32,197,6,70,227,208,3,32,202
    ,6,202,208,212,198,235,208,206,198,236
    ,240,202
MR  1100 DATA 32,224,6,198,218,208,188,96,
    102,226,230,229,96,102,227,230,216,208
    ,2,230,217
XZ  1110 DATA 96,165,220,133,235,165,221,1
    33,236,165,233,133,226,96,24,165,203,1
    01,232,133,203
NY  1120 DATA 165,204,105,0,133,204,96,169
    ,0,133,228,56,165,224,229,231,144
GU  1130 DATA 6,133,224,230,228,208,243,16
    4,225,240,8,133,224,198,225,230,228,20
    8,231,165,224
GJ  1140 DATA 133,233,165,234,201,1,240,8,
    164,234,6,233,136,136,208,250,166,233,
    189,112,4
IK  1150 DATA 133,226,133,233,169,128,133,
    227,166,222,202,224,255,240,6,32,224,6
    ,24,144,245
GV  1160 DATA 230,218,230,220,208,2,230,22
    1,96,4,8,4,8,4,8,2,2,2,8,4,10
LF  1170 DATA 10,20,20,40,40,40,40,40,20,4
    0,2,1,2,1,2,1,4,4,4,1,2
DZ  1180 DATA 128,64,32,16,8,4,2,1
```

## BASIC VARIABLE RENAMING TOOL

# VRENAME

*Article on page 12*

## *LISTING 1*

Don't type the TYPO II Codes!

```
HE  0 GOTO 31500
ME  31500 REM VARIABLE RENAMING UTILITY
OL  31501 REM BY DOUG WHITE
EB  31502 REM (C)1989 ANTIC PUBLISHING INC
OF  31504 ? "▚":POSITION 2,2
ZC  31505 ? "     VARIABLE RENAMING UTILITY
    "
NH  31506 ? "            BY DOUG WHITE"
CE  31507 ?
DF  31508 CLR
KO  31510 DIM VARNAME$(2000),NAME$(128)
WV  31512 DIM YN$(1)
II  31514 VNT=PEEK(130)+256*PEEK(131)
EJ  31516 N=1:FLAG=0
NQ  31518 REM ... GET NEXT VARIABLE NAME
WJ  31520 BYTE=PEEK(VNT)
LD  31522 NAME$(N,N)=CHR$(BYTE)
AX  31524 REM .. 0 INDICATES END OF TABLE
ZR  31526 IF BYTE=0 THEN 31602
EQ  31528 REM ... CHECK FOR INVERSE CHAR.
NB  31530 IF BYTE<128 THEN VNT=VNT+1:N=N+1
    :GOTO 31520
DM  31532 NAME$(N,N)=CHR$(BYTE-128)
HM  31534 IF NAME$="VARNAME$" THEN 31602
LP  31536 ? :? NAME$:?
AN  31538 ? "DO YOU WANT TO CHANGE THIS VA
    RIABLE NAME (Y/N) ";
MC  31540 INPUT YN$
PJ  31542 IF YN$="Y" THEN 31558
UL  31544 REM ... PUT OLD NAME IN VNAME$
UZ  31546 NLEN=LEN(NAME$):VLEN=LEN(VARNAME
    $)
MP  31548 NAME$(N,N)=CHR$(BYTE)
BP  31550 VARNAME$(VLEN+1,VLEN+1+NLEN)=NAM
    E$
CZ  31552 VNT=VNT+1:N=1:NAME$=" "
CU  31554 GOTO 31520
DT  31556 REM
RE  31558 FLAG=1:REM .. DENOTES A CHANGE
UF  31560 NAME$=" "
OC  31562 ? :? "  INPUT NEW VARIABLE NAME"
    :?
UT  31564 ? "N THE LAST CHARACTER OF A STR
    ING NAME"
BU  31566 ? "O MUST BE '$'"
JZ  31568 ? "T THE LAST CHARACTER OF AN AR
    RAY NAME"
KH  31570 ? "E MUST BE '('":?
MC  31572 INPUT NAME$
FM  31574 NLEN=LEN(NAME$)
EQ  31576 IF BYTE=164 AND NAME$(NLEN,NLEN)
    <>"$" THEN 31558
OC  31578 IF BYTE=168 AND NAME$(NLEN,NLEN)
    <>"(" THEN 31558
LF  31580 ? :? NAME$:?
GN  31582 ? "IS THIS CORRECT (Y/N) ";
NM  31584 INPUT YN$
KK  31586 IF YN$="N" THEN 31558
FB  31588 REM ... PUT NEW NAME IN VNAME$
OP  31590 NLEN=LEN(NAME$):VLEN=LEN(VARNAME
    $)
BH  31592 NAME$(NLEN,NLEN)=CHR$(ASC(NAME$(
    NLEN,NLEN))+128)
CZ  31594 VARNAME$(VLEN+1,VLEN+1+NLEN)=NAM
    E$
EJ  31596 VNT=VNT+1:N=1:NAME$=" "
WI  31598 GOTO 31520:REM ..GET NEXT NAME
BY  31600 REM
RB  31602 IF FLAG=0 THEN ? "NO CHANGES MAD
    E":END
CS  31604 REM
MD  31606 REM ... STORE OLD POINTERS
LO  31608 VNT=PEEK(130)
AE  31610 VNT1=PEEK(131)
```

*continued on next page*

```
BR 31612 VNT2=PEEK(132)
DE 31614 VNT3=PEEK(133)
KK 31616 TRAP 31698
UA 31618 VLEN=LEN(VARNAME$)
LN 31620 VARNAME$(VLEN+1,VLEN+1)=CHR$(0)
BA 31622 REM ... MAKE VNTP & VNTD
TO 31624 REM ... POINT TO VNAMES
DQ 31626 VNADR=ADR(VARNAME$)
MX 31628 HIBYTE=INT(VNADR/256)
RX 31630 LOWBYTE=VNADR-256*HIBYTE
EH 31632 REM ... POKE HIBYTE & LOWBYTE
SN 31634 REM ... INTO VNTP
GA 31636 POKE 130,LOWBYTE:POKE 131,HIBYTE

LW 31638 VNEND=VNADR+LEN(VARNAME$)
KZ 31640 HIBYTE=INT(VNEND/256)
RT 31642 LOWBYTE=VNEND-256*HIBYTE
EV 31644 REM ... POKE HIBYTE & LOWBYTE
IL 31646 REM ... INTO VNTD
KA 31648 POKE 132,LOWBYTE:POKE 133,HIBYTE

CS 31650 REM
OS 31652 ? "▚":POSITION 2,2
QK 31654 ? "INPUT THE NAME OF THE DISK FI
   LE"
MF 31656 ? "THAT YOU WANT TO USE FOR THE
   MODIFIED"
VP 31658 ? "PROGRAM.  '.LST' WILL BE ADDE
   D FOR YOU"
ZJ 31660 ? "     EXAMPLE:  D:FILENAME":?
CE 31662 NAME$=""
ML 31664 INPUT NAME$
PJ 31666 IF LEN(NAME$)=0 THEN 31652
FB 31668 IF NAME$(2,2)=":" AND LEN(NAME$)
   <11 THEN 31678
GO 31670 IF NAME$(3,3)=":" AND LEN(NAME$)
   <12 THEN 31678
JA 31672 ? :? "THE NAME YOU CHOSE IS TOO
   LONG"
LP 31674 ? "PLEASE INPUT A SHORTER NAME"
GL 31676 GOTO 31660
BL 31678 NAME$(LEN(NAME$)+1)=".LST"
XI 31680 LIST NAME$,1,31499
GH 31682 GRAPHICS 0:POSITION 2,4
YA 31684 ? "ENTER ";CHR$(34);NAME$;CHR$(3
   4)
NR 31686 POSITION 2,1
TK 31688 ? "PRESS RETURN"
FS 31690 NEW
ZW 31692 END
RF 31694 REM ... ERROR TRAP ROUTINE
WF 31696 REM ... RESTORE OLD POINTERS
EV 31698 ? :? "OLD NAMES RESTORED"
TK 31700 POKE 130,VNT
ED 31702 POKE 131,VNT1
FU 31704 POKE 132,VNT2
HL 31706 POKE 133,VNT3
ZI 31708 END
```

## LISTING 2

```
BL 10 REM SHELL SORT
JK 15 REM BY DOUG WHITE
LB 17 REM (C)1989 ANTIC PUBLISHING INC.
TL 20 DIM A(1000)
LR 30 GRAPHICS 0
OJ 40 ? "       SHELL SORT"
UC 50 ? :? "HOW MANY #'S ARE YOU SORTING
   (< 1000)"
HM 60 INPUT NUM
QA 70 IF NUM>1000 THEN 50
HM 80 ? :? "FILLING THE ARRAY WITH UNSORT
   ED #'S"
TT 90 FOR N=1 TO NUM
NX 100 A(N)=INT(10000*SIN(N))
HR 110 NEXT N
BX 120 ? :? " ***  SORTING  ***"
EF 130 POKE 18,0:POKE 19,0:POKE 20,0
QU 140 REM
MB 500 REM ..... BEGINNING OF SHELL SORT
KG 510 PSIZE=1
ZH 520 PSIZE=PSIZE*3+1
JB 530 IF PSIZE<NUM THEN 520
QY 540 REM
VZ 550 PSIZE=INT(PSIZE/3)
EB 560 ? "PARTITION SIZE = ",PSIZE
RE 570 REM
BF 580 FOR N=1+PSIZE TO NUM
FW 590 TEMP=A(N)
MD 600 K=N-PSIZE
NL 610 FOR J=K TO 1 STEP -PSIZE
HT 620 IF TEMP<A(J) THEN 650
AN 630 A(J+PSIZE)=A(J)
GO 640 NEXT J
JX 650 A(J+PSIZE)=TEMP
IG 660 NEXT N
RB 670 IF PSIZE>1 THEN 550
PS 680 REM ..... END OF SHELL SORT
HW 1000 REM
OA 1010 T=PEEK(20)+256*PEEK(19)
RI 1020 SOUND 3,100,10,2
UJ 1030 FOR N=1 TO NUM:? N,A(N):NEXT N
WZ 1040 ? :? T/60;" SECONDS "
FC 1050 END
```

---

### THE GREAT PASTA SAUCE CHASE . . . DON'T ASK!

# GOOD KING ZURP

*Article on page 17*

## LISTING 1

Don't type the TYPO II Codes!

```
XZ 0 REM GOOD KING ZURP
GN 1 REM BY FRANK MARTONE
FF 2 REM (C)1989, ANTIC PUBLISHING INC.
DW 3 DIM F$(13):GRAPHICS 17:POKE 708,15:S
   N=0:POSITION 3,10
PU 4 ? #6;"ANTIC PRESENTS":GOSUB 1000:GOS
   UB 3000:GOSUB 400
PC 5 GRAPHICS 17:TUM=9:SC=0:BRS=30:DIFF=4
   00:S1=19:S2=20:CTT=0:BI=5:S3=1:S4=23
XM 6 POKE 756,CH/256:S5=0:S6=3:EG=10:GOB=
   7:SCR=PEEK(88)+256*PEEK(89)
AF 7 GOSUB 11111:POKE 711,255:POKE 708,13
   8
XB 8 FOR G=2 TO 7:POSITION 0,G:? #6;"cccc
   cccccccccccccccc":SOUND 0,G+6,10,7-G:N
   EXT G
BP 9 FOR G=8 TO 22:POSITION 0,G:? #6;"▀▀▀▀
   ▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀":SOUND 0,G,10,10:SOU
   ND 0,0,0,0:NEXT G
SJ 11 FOR B=1 TO BRS:K1=INT(RND(0)*19):K2
   =INT(RND(0)*20)+3:POSITION K1,K2
ZM 12 ? #6;"▚":SOUND 0,20,10,10:FOR HJ=0
   TO 1:NEXT HJ:SOUND 0,0,0,0:NEXT B
FM 13 FOR T=1 TO EG:E1=INT(RND(0)*19):E2=
   INT(RND(0)*20)+3:POSITION E1,E2
JO 14 ? #6;";":SOUND 0,120,10,10:FOR HJ=1
    TO 3:NEXT HJ:SOUND 0,0,0,0:NEXT T
CY 15 FOR C=1 TO GOB:POSITION RND(0)*18,R
   ND(0)*18+3:? #6;"j":SOUND 0,C,10,10:SO
   UND 0,0,0,0:NEXT C
ZE 16 X=10:Y=12:V=10:TIME=0:FOR G=10 TO 1
   3:POSITION 9,G:? #6;"   ":NEXT G
TK 17 POKE SCR+X+20*Y,10:FOR G=1 TO 80:SO
   UND 0,G*2,0,10:NEXT G:SOUND 0,0,0,0:PO
   SITION 16,1:? #6;"+":TIME=30
XL 18 POSITION 0,0:? #6;"score   time   li
   ves":COLOR 32:PLOT X,Y
QK 20 POSITION 9,1:? #6;INT(TIME);" ";:GO
```

```
   SUB 4100
JT 21 ST=PEEK(632):TR=PEEK(644):TP=SCR+X+20*Y:POKE 709,14:POKE 77,0
MI 25 IF TIME<11 THEN SOUND 1,TIME+30,10,10
WO 31 POSITION 1,1:? #6;SC:POSITION 17,1:? #6;TUM
YL 32 LOCATE X,Y,BC
UN 33 IF BC=35 THEN FOR G=1 TO 4:FOR R=-15 TO 15 STEP 3:SOUND 0,ABS(R)+35,10,10:NEXT R:NEXT G:SC=5
AM 35 SOUND 1,0,0,0:TIME=TIME-0.5:V=V+1:IF TIME<1 THEN 7000
ON 47 IF BC=247 THEN FOR W=15 TO 0 STEP -2:SOUND 0,30,10,W:NEXT W:SC=SC+100
FB 48 IF BC=59 THEN FOR W=15 TO 0 STEP -2:SOUND 0,W*8,10,W:NEXT W:SC=SC+250
TW 49 IF BC=106 THEN FOR W=70 TO 60 STEP -2:SOUND 0,RND(0)*5+3,10,8:NEXT W:SC=SC+500
GY 50 POKE SCR+X+20*Y,V:IF BC=40 THEN GOSUB 300
YS 51 POKE 709,0:IF X>18 THEN POKE SCR+X+20*Y,0:X=X-1
QK 52 IF X<1 THEN POKE SCR+X+20*Y,0:X=X+1
UD 53 IF Y<3 THEN POKE SCR+X+20*Y,0:Y=Y+1
IO 54 IF Y>22 THEN POKE SCR+X+20*Y,0:Y=Y-1
OY 55 IF RND(0)*6<4 THEN GOSUB 500
MP 57 IF RND(0)*5<2 AND SC>15000 THEN GOSUB 550
WB 58 IF RND(0)*4<2 AND SC>29000 THEN GOSUB 580
JY 60 SOUND 0,0,0,0:IF BC=45 THEN GOSUB 90
JD 61 IF X=S1 AND Y=S2 THEN GOSUB 300
LL 62 IF X=S3 AND Y=S4 THEN GOSUB 300
NT 63 IF X=S5 AND Y=S6 THEN GOSUB 300
NP 64 IF SC>60000 THEN POKE 710,PEEK(53770)
DG 65 IF SC>39000 AND PEEK(53770)=44 THEN GOSUB 100
QF 66 IF SC>100000 THEN POKE 710,0:FOR D=1 TO 1:NEXT D
NA 80 IF V>11 THEN V=10
IR 81 POKE 709,0:POKE 709,14:IF RND(0)*DIFF<13 THEN POSITION RND(0)*19,RND(0)*18+3:? #6;"-":SOUND 0,255,10,5
SC 82 GOTO 20
YW 90 FOR R=1 TO 4:POSITION X,Y:? #6;"-":FOR D=1 TO 10:NEXT D:SOUND 0,255,10,10
FZ 91 POSITION X,Y:? #6;" ":FOR D=1 TO 10:NEXT D:SOUND 0,0,0,0:NEXT R:GOSUB 300
AH 94 RETURN
AQ 100 POSITION RND(0)*19,RND(0)*18+4:? #6;"#"
ZO 101 FOR R=2 TO 20:SOUND 0,R*4,10,10:SOUND 1,R*8,10,10:POKE 712,R*5:SOUND 2,R,10,10:POKE 712,R:NEXT R
WT 102 POKE 712,0:FOR T=0 TO 3:SOUND T,0,0,0:NEXT T:RETURN
RJ 300 POSITION X,Y:? #6;"+":FOR G=14 TO 1 STEP -1:FOR W=1 TO 25:NEXT W:POKE 708,64+G
HR 301 SOUND 0,G,10,G:SOUND 1,G+2,10,G:NEXT G:POKE 708,206:SOUND 0,0,0,0:SOUND 1,0,0,0
NM 302 FOR EX=4 TO 6:POKE SCR+X+20*Y,EX:FOR W=1 TO 30:NEXT W:SOUND 0,EX*4,10,10:NEXT EX:POKE SCR+X+20*Y,0:X=10:Y=12
QO 303 FOR S=0 TO 2:SOUND S,0,0,0:NEXT S
XM 304 TUM=TUM-1
RI 305 IF TUM=0 THEN GOTO 5000
OY 306 FOR L=15 TO 1 STEP -0.3:SOUND 1,L,6,L:SOUND 2,L,6,L:NEXT L
NP 310 FOR R=0 TO 2:SOUND R,0,0,0:NEXT R
AW 311 POSITION S1,S2:? #6;" ":POSITION S3,S4:? #6;" ":POSITION S5,S6:? #6;" "
HX 312 S1=3:S2=19:S3=18:S4=19:S5=3:S6=3:POKE 708,138
ZD 320 RETURN
UL 400 GRAPHICS 17:POKE 711,137:S1=19:S2=18:S3=2:S4=18:S5=0:S6=5:POKE 756,CH/256:POKE 709,14
MM 405 DL=PEEK(560)+256*PEEK(561):POKE DL+13,7:POKE DL+15,7
LN 409 POSITION 3,7:? #6;"j j j j j j j j j"
LY 410 POSITION 3,8:? #6;"entering screen":SN=SN+1
KD 411 POSITION 10,10:? #6;SN
QO 412 POSITION 3,11:? #6;"j j j j j j j j"
LZ 417 FOR G=1 TO 2
OQ 420 FOR D=15 TO 0 STEP -1:SOUND 0,34,10,D:POKE 708,D:NEXT D:NEXT G:RETURN
FQ 500 SNK=SCR+S1+20*S2:SV=58
WK 510 POKE SNK,0:SOUND 0,20,6,5
ZL 521 IF Y<S2 AND X=S1 THEN S2=S2-1:GOTO 529
XU 522 IF Y>S2 AND X=S1 THEN S2=S2+1:GOTO 529
TT 525 IF X<S1 THEN S1=S1-1:SV=58
IK 526 IF X>S1 THEN S1=S1+1:SV=7
VT 529 POKE SCR+S1+20*S2,SV:RETURN
RH 550 SNK1=SCR+S3+20*S4:SV=58
FT 555 POKE SNK1,0:SOUND 0,10,6,5
YF 557 IF X<S3 THEN S3=S3-1:SV=58
MW 558 IF X>S3 THEN S3=S3+1:SV=7
XG 559 IF Y<S4 AND X=S3 THEN S4=S4-1
UN 560 IF Y>S4 AND X=S3 THEN S4=S4+1
XX 561 POKE SCR+S3+20*S4,SV:RETURN
UP 580 SNK2=SCR+S5+20*S6:SV=58
EX 585 POKE SNK2,0:SOUND 0,3,6,5
CL 587 IF X<S5 THEN S5=S5-1:SV=58
RC 588 IF X>S5 THEN S5=S5+1:SV=7
EK 589 IF Y<S6 AND X=S5 THEN S6=S6-1
BR 590 IF Y>S6 AND X=S5 THEN S6=S6+1
AX 591 POKE SCR+S5+20*S6,SV:RETURN
CX 1000 REM --CHARACTER SET--
YK 1010 CH=(PEEK(106)-8)*256:FOR I=0 TO 512:POKE CH+I,PEEK(57344+I):NEXT I
HN 1020 RESTORE 1100
SX 1030 READ A:IF A<0 THEN RETURN
XD 1040 FOR J=0 TO 7:READ B:POKE CH+A*8+J,B:NEXT J:GOTO 1030
XD 1100 DATA 10,0,189,126,235,255,255,126,36
KC 1110 DATA 11,0,36,126,255,106,255,126,60
WM 1115 DATA 12,0,60,126,126,255,106,126,189
SC 1120 DATA 4,0,0,4,0,16,8,64,0
KM 1125 DATA 5,1,66,0,34,8,64,20,128
JU 1130 DATA 6,0,0,4,80,8,20,0,0
JL 1140 DATA 2,255,253,239,247,255,191,251,191
FM 1143 DATA 58,48,88,56,8,18,34,36,24
VL 1146 DATA 55,0,15,31,62,124,120,112,0
RT 1149 DATA 63,7,5,7,56,40,184,128,128
VA 1150 DATA 42,56,84,146,254,170,84,40,16
DM 1170 DATA 3,146,84,16,238,16,84,146,0
XK 1175 DATA 26,0,0,0,0,0,0,1,3
MT 1176 DATA 27,0,126,255,171,171,255,255,0
LJ 1177 DATA 28,0,0,0,0,0,0,128,192
DQ 1178 DATA 29,7,7,7,15,15,15,15,7
KF 1179 DATA 30,255,156,8,8,207,8,156,255
TA 1180 DATA 31,224,224,112,112,112,112,240,224
KU 1181 DATA 32,7,3,7,24,48,24,4,0
MF 1182 DATA 61,189,195,255,126,60,24,24,0
YU 1183 DATA 62,224,192,224,24,12,24,32,0
HD 1184 DATA 49,0,0,0,255,0,255,0,0
NQ 1190 DATA 8,170,77,170,77,170,77,170,77
UN 1191 DATA 13,254,170,170,170,170,170,170,254
AB 1192 DATA 7,6,13,14,8,68,66,34,28
WM 1193 DATA 8,3,192,12,0,51,0,6,96,-1
IW 3000 GRAPHICS 17:POKE 756,CH/256:POKE 708,75:POKE 711,15:POKE 709,11
PZ 3003 POKE 710,255:POSITION 4,7:? #6;"jjjj jjjj jjjj"
TU 3026 POSITION 9,9:? #6;"BAK":POSITION 9,10:? #6;"⊟⊠":POSITION 9,11:? #6;"@J^"
HF 3040 COLOR 81:PLOT 0,22:DRAWTO 19,22:PLOT 0,3:DRAWTO 19,3
BR 3041 L=0.2:POSITION 0,3:? #6;"QQQQQQQQQQQQQQQQQQQQ"
SL 3099 RESTORE 3200:R=4
OT 3100 READ F$:IF F$="X" THEN GOTO 3500
```

*continued on next page*

```
GP 3101 POSITION R,7:? #6;"%":FOR D=15 TO
   0 STEP -1.3:SOUND 0,D+10,8,D:NEXT D
XC 3102 POSITION R,7:? #6;F$;
YM 3105 R=R+1:GOTO 3100
NQ 3200 DATA G,O,O,D,j,K,I,N,G,j,Z,U,R,P,
   X
CJ 3500 POSITION 2,16:? #6;" BY FRANK MAR
   TONE":SOUND 0,0,0,0:SOUND 1,0,0,0
YN 3503 FOR G=1 TO 100:FOR J=15 TO 0:SOUN
   D 0,J*2,0,R:SOUND 1,J*3,0,J
GD 3505 POKE 709,J+100:NEXT J:NEXT G:SOUN
   D 0,0,0,0:SOUND 1,0,0,0
BK 4000 GRAPHICS 17:POKE 756,CH/256:POKE
   708,14:POKE 710,55:POKE 709,255
YW 4001 P=5:CC=0:POSITION 4,2:? #6;" poin
   t scale"
LQ 4002 COLOR 81:PLOT 0,3:DRAWTO 19,3:PLO
   T 0,22:DRAWTO 19,22
NO 4004 POSITION 1,P:? #6;"PASTA PKT    w
    100":GOSUB 4900
NG 4005 POSITION 1,P:? #6;"SEASONINGS  ▒
    250":GOSUB 4900
MN 4006 POSITION 1,P:? #6;"GARLIC      ▒
    500":GOSUB 4900
ND 4007 POSITION 1,P:? #6;"SPEC. PASTA #
    5000":GOSUB 4900
JG 4008 POSITION 1,17:? #6;"AVOID COLLAND
   ERS ▒"
LP 4009 POSITION 1,19:? #6;"AVOID MAGGOTS
    ▒'"
OJ 4010 POSITION 5,21:? #6;"PRESS START":
   FOR D=1 TO 50:NEXT D
MQ 4011 POSITION 5,21:? #6;"press start":
   FOR D=1 TO 50:NEXT D:CC=CC+1:IF CC=30
   THEN GOTO 3000
BR 4015 IF PEEK(53279)=6 THEN GOTO 5
OR 4020 GOTO 4010
WO 4100 POKE TP,0:X=X+(ST<8)-(ST>8 AND ST
   <12):Y=Y+(ST=13)-(ST=14):RETURN
PC 4900 FOR D=15 TO 0 STEP -1:SOUND 0,50,
   10,D:NEXT D:P=P+3:RETURN
RI 5000 POSITION 17,1:? #6;"0":FOR E=14 T
   O 0 STEP -1:POKE 710,64+E
BY 5001 FOR R=1 TO 10:NEXT R:SOUND 0,E,0,
   10:NEXT E:SC=0:SN=0
CF 5005 FOR D=7 TO 12:POSITION 5,D:? #6;"
        ":NEXT D
RK 5010 POSITION 6,8:? #6;"game over"
AQ 5050 FOR L=20 TO 70:SOUND 0,L,10,10:SO
   UND 1,L,6,10:NEXT L
SO 5061 SOUND 0,0,0,0:SOUND 1,0,0,0:C=0
YQ 5065 C=C+1:POSITION 5,10:? #6;"PRESS S
   TART":C=C+1
BT 5076 IF C=28 THEN FOR I=19 TO 0 STEP -
   1:COLOR 0:PLOT I,0:DRAWTO I,23:FOR D=1
    TO 5:NEXT D:NEXT I:GOTO 3000
LR 5080 IF PEEK(53279)=6 THEN SN=0:GOTO 5
UD 6000 GOTO 5065
NM 7000 COLOR 176:PLOT 9,1:COLOR 42:PLOT
   X,Y
VE 7001 SOUND 0,255,10,10:SOUND 1,254,10,
   10:FOR L=1 TO 50:POKE 710,PEEK(53770):
   NEXT L
LB 7002 RESTORE 7100
WH 7005 READ MUSIC:IF MUSIC=255 THEN 7120
LE 7010 SOUND 0,MUSIC,10,10:FOR G=7 TO 1
   STEP -1.5:SOUND 1,MUSIC,10,G+4
PE 7025 NEXT G:POKE 712,MUSIC:POKE 710,PE
   EK(53770):GOTO 7005
BG 7100 DATA 121,1,121,121,96,114,121,96,
   96,1,96,96,81,91,96,91,91,1,91,91,76,8
   1,91,96,96,114,114,121,121,121,255
LR 7120 SOUND 0,0,0,0:SOUND 1,0,0,0:POKE
   712,0:FOR I=0 TO 19:COLOR 0
SO 7122 PLOT I,0:DRAWTO I,23:FOR D=1 TO 5
   :NEXT D:NEXT I
ZL 7201 IF DIFF<110 THEN DIFF=120
YJ 7202 CTT=CTT+1
QC 7204 IF CTT=3 THEN 9000
OU 7207 EG=EG+7:WB=WB+0.5:DIFF=DIFF-27:GO
   SUB 400:BRS=BRS+1
ND 7210 GRAPHICS 17:POKE 756,CH/256:POKE
   710,PEEK(53770):POKE 708,138:GOB=GOB+7
   :GOTO 7
NS 9000 GRAPHICS 17:POKE 710,14:POKE 756,
   CH/256:POSITION 5,6:? #6;"bonus round"
   :CTT=0
JU 9001 POSITION 3,10:? #6;"YOU MUST GRAB
    3"
NM 9002 POSITION 3,12:? #6;"SPECIAL PASTA
   S"
BO 9003 POSITION 4,14:? #6;"TO GET BONUS"
RC 9004 COLOR 33:PLOT 0,16:DRAWTO 19,16:P
   LOT 0,4:DRAWTO 19,4
MN 9006 FOR D=1 TO 140:POKE 708,RND(0)*10
   +50:NEXT D
WL 9010 GRAPHICS 17:GOSUB 11111:POKE 756,
   CH/256:POSITION 5,0:? #6;"bonus round"
   :X=10:Y=10:POKE 708,156
LW 9011 POSITION 0,22:? #6;"aaaaaaaaaaaaa
   aaaaaaa":TIME=0:T=10:IF SC>30000 THEN
   T=6:S1=4:S2=16
KI 9012 POSITION 0,3:? #6;"aaaaaaaaaaaaaa
   aaaaaa":IF SC>70000 THEN T=3
PT 9013 POSITION 5,23:? #6;"AVOID TRAPS"
GM 9014 FOR K=5 TO 21 STEP 3
DY 9015 POSITION 3,K:? #6;" -   -   -   -
   "
DQ 9017 NEXT K:SCR=PEEK(88)+256*PEEK(89):
   BS=0
WC 9050 TP=SCR+X+20*Y:ST=PEEK(632):POKE 7
   10,14
OW 9051 IF X=S1 AND Y=S2 THEN 9700
AD 9053 IF SC>50000 AND RND(0)*9<5 THEN G
   OSUB 500
NI 9055 GOSUB 4100:IF X>18 THEN POKE SCR+
   X+20*Y,0:X=X-1
SU 9060 IF X<1 THEN POKE SCR+X+20*Y,0:X=X
   +1
XV 9061 IF Y<5 THEN POKE SCR+X+20*Y,0:Y=Y
   +1
OQ 9062 IF Y>20 THEN POKE SCR+X+20*Y,0:Y=
   Y-1
BY 9063 SOUND 0,0,0,0:IF BS=3 THEN 9200
LU 9070 POKE TP,11:FOR D=1 TO 8:NEXT D:LO
   CATE X,Y,BC:POSITION 7,2:? #6;"COUNT "
   ;BS
CW 9075 IF BC=45 THEN 9700
RW 9080 POKE TP,0:POKE 710,PEEK(53770)
JP 9100 IF TIME=0 THEN T1=INT(RND(0)*15)+
   3:T2=INT(RND(0)*15)+4:POSITION T1,T2:?
    #6;"▒"
KJ 9103 TIME=TIME+1:IF TIME>T THEN POSITI
   ON T1,T2:? #6;" ":FOR D=10 TO 0 STEP -
   1:SOUND 0,10,10,D:NEXT D:TIME=0
YB 9107 IF BC=163 THEN FOR D=1 TO 10:SOUN
   D 0,D+2,8,6:POKE 712,D*3:NEXT D:BS=BS+
   1:SOUND 0,0,0,0:POKE 712,0
TK 9150 GOTO 9050
II 9200 REM
KF 9240 POSITION 3,23:? #6;"CONGRATULATIO
   NS"
VN 9241 POSITION 5,2:? #6;"RANdoM BOnUS "
SN 9245 FOR D=1 TO 10:SOUND 0,RND(0)*10,1
   0,10:SOUND 1,RND(0)*20,10,10:NEXT D:A=
   INT(RND(0)*10)*1000+1000:SC=SC+A
SP 9246 POSITION 5,2:? #6;"BONUS ";A;"
   ":SOUND 0,0,0,0:SOUND 1,0,0,0
YT 9247 FOR D=1 TO 10:FOR F=15 TO 0 STEP
   -1:POKE 708,F:NEXT F:NEXT D
CK 9248 SN=SN+1:GOSUB 400:GRAPHICS 17:POK
   E 756,CH/256:POKE 710,121:POKE 708,138
   :GOTO 7
LA 9700 POKE TP,0:POSITION 3,23:? #6;"SOR
   RY NO BONUS"
QT 9704 FOR JJ=30 TO 50
YH 9705 FOR D=-15 TO 15 STEP 3:SOUND 0,AB
   S(D)+JJ,10,10:POKE 708,D+50:NEXT D
OY 9706 NEXT JJ
DZ 9707 SOUND 0,20,6,10:POKE 708,255:FOR
   D=1 TO 40:NEXT D:SOUND 0,0,0,0
SD 9710 SN=SN+1:GOSUB 400:GRAPHICS 17:POK
   E 756,CH/256:POKE 710,137:POKE 708,138
   :GOTO 7
VW 9999 GOTO 9050
PI 10000 REM --DLI-BLUE--
KD 11111 RESTORE 11115
UX 11112 FOR ADD=1536 TO 1536+28:READ B:P
   OKE ADD,B:NEXT ADD
SG 11115 DATA 72,138,72,141,10,212,169,11
   2,141,26,208,162
LG 11116 DATA 15,141,10,212,202,208,250,1
   73,200,2,141,26,208,104,170,104,64
RC 11120 POKE 512,0:POKE 513,6:DL=PEEK(56
   0)+256*PEEK(561)
ID 11130 POKE DL+2,112+128:POKE 54286,192
VN 11147 POSITION 6,0:? #6;"get ready":RE
   TURN
```

# Tech Tips

## THREE COLORFUL GRAPHICS DEMOS

*By Mark Proudfoot*

These three colorful BASIC graphics demonstrations were sent by Mark Proudfoot, 14, a high school student from Brentwood, California.

**Easter Egg Maker** produces a beautiful, flashing Easter egg, using pastel shades for excellent effect. **Rainbow** is a Graphics 11 demo with sound, producing a different set of "rainbows" each time. **Pipe** is in Graphics 9 and produces a pipe with water running out of it. The delicate shading gives an almost three-dimensional effect.

Don't type the TYPO II Codes!

## LISTING 1

```
FQ 1 REM EASTER EGG MAKER
OF 2 REM BY MARK PROUDFOOT
TO 3 REM (C)1989, ANTIC PUBLISHING INC
QH 10 GRAPHICS 7:POKE 752,1:POKE 708,14:C
   OLOR 1:POKE 710,208:POKE 712,208:POKE
   709,0
CV 12 ? "The Easter Egg Maker-by Mark Pro
   udfoot  Press START to create another
   egg."
US 13 ? "            Press SELECT to end.":FO
   R F=1 TO 2
TW 15 R=1.1:IR=1/R:PI=6.28318531:NPT=360
DG 20 NX=80:NY=40:SIZE=42/R:DX=R*PI/NPT:D
   Y=IR*PI/NPT
PB 35 X=0:Y=SIZE
AC 40 FOR I=1 TO 80 STEP 1
YM 50 PLOT X+NX,Y+NY:PLOT NX+X,NY-Y:PLOT
   NX-X,NY+Y:GOSUB 80:PLOT NX-X,NY-Y:GOSU
   B 80:IF F=2 THEN GOSUB 200
KA 60 X=X+(Y*(DX)):Y=Y-1.4*(X*(DY))
BH 70 TRAP 130:NEXT I:NEXT F:GOTO 190
QE 80 IF F=1 THEN RETURN
QZ 90 GOSUB 160
TY 92 SETCOLOR 1,RND(0)*15,14
SM 100 POKE 765,RND(0)*3
VT 110 XIO 18,#6,0,0,"S"
ZB 120 RETURN
PB 130 F=2:GOTO 15
QW 150 REM
BQ 160 SETCOLOR 0,RND(0)*15,14
CG 170 SETCOLOR 1,RND(0)*15,14
ZN 180 RETURN
CO 190 GOSUB 160:GOSUB 200:GOTO 190
DO 200 IF PEEK(53279)=6 THEN RUN
NV 210 IF PEEK(53279)=5 THEN GRAPHICS 0:E
   ND
ZC 220 RETURN
```

## LISTING 2

```
RW 1 REM RAINBOW
OF 2 REM BY MARK PROUDFOOT
TO 3 REM (C)1989, ANTIC PUBLISHING INC
UR 10 GRAPHICS 11:F=3:POKE 712,13
SI 20 GOSUB 100:IF G=3 THEN GOTO 130
YD 30 REM VERTICAL LINES
DD 40 PLOT 0,RND(0)*179
SQ 50 FOR B=1 TO 79 STEP 2:C=RND(0)*179:C
   OLOR INT(RND(0)*14)+2
SK 60 DRAWTO B,C:SOUND 1,C,0,10
GK 70 NEXT B
GR 80 F=6:GOTO 20
JQ 90 REM HORIZONTAL LINES
KA 100 G=G+1:FOR A=1 TO 185 STEP F:COLOR
   INT(A/15)+1:IF G=2 THEN COLOR 13
MF 110 SOUND 0,0,0,A/15:PLOT 0,A:DRAWTO 7
   9,A
SZ 120 NEXT A:COLOR 0:RETURN
AS 130 SOUND 1,0,0,0:FOR DE=1 TO 500:NEXT
    DE
MG 140 COLOR 0:FOR A=185 TO 1 STEP -2
YT 150 SOUND 0,0,0,A/15:PLOT 0,A:DRAWTO 7
   9,A:NEXT A
RV 160 GOSUB 210
LD 170 FOR A=15 TO 0 STEP -0.5
FK 180 SOUND 0,0,0,A:POKE 712,A
DH 190 NEXT A
AT 200 PRINT #6;"K":GOSUB 210:END
HY 210 FOR DE=1 TO 1000:NEXT DE:RETURN
```

## LISTING 3

```
PA 5 REM PIPE
UJ 6 REM BY MARK PROUDFOOT
TS 7 REM (C)1989, ANTIC PUBLISHING INC
YR 10 GRAPHICS 9:POKE 712,16:DEG
HN 40 FOR B=1 TO 61
PZ 50 FOR A=10 TO 17:COLOR A-9
JT 60 PLOT A,B:NEXT A
TV 70 FOR A=18 TO 25:COLOR 25-(A-4)
TS 80 PLOT A,B:NEXT A:NEXT B
DJ 90 FOR A=31 TO 46:K=A-30:COLOR K
JU 100 PLOT 25-INT(K/3),A:DRAWTO 40,A
PP 110 NEXT A:C=15
CX 115 FOR A=46 TO 61:K=ABS(A-61):COLOR K
JY 120 PLOT 25-INT(K/3),A:DRAWTO 40,A
AF 125 NEXT A:C=15:SOUND 1,25,0,2
TO 130 X=41:Y=INT(RND(0)*3)+58:G=(RND(0)*
   2)+2
HE 135 SOUND 0,PEEK(53770),10,2:SOUND 0,P
   EEK(53770),8,2
DQ 140 COLOR (RND(0)*8)+1:G=G*1.5:PLOT X,
   Y
MK 150 X=X+2:Y=Y+G
BU 160 IF X>78 OR Y>179 THEN GOTO 130
PJ 170 GOTO 135
```

*Antic pays $25 for every original and exclusive Tech Tip submission that we publish. Send your 8-bit or ST disk and printout to: Antic Tech Tips, 544 Second Street, San Francisco, CA 94107. Tech Tips welcomes very short programs that demonstrate the Atari's powers, simple hardware modifications, or useful macros for popular software.*

# Antic Classifieds

★★★★★★

**Munich Atari XL/XE** Public Domain Association is looking for members. Great PD-selection. We sell ($2/2S), change and buy PD-software. Free list. Write to: Florian Bechert, August-Foppl-Str.17 , 8000 Munchen 50, West-Germany (FRG)(8/89)

★★★★★★

Try us for your Atari Public Domain software needs. Good prices, fast service. Write for free catalog. Vulcan Software, P.O. Box 692 Manassas, VA 2211-0692 (9/89)

PRINTSHOP GRAPHICS. Holidays $15.00. LDS Prophets and Temples $15.00. LDS Church Auxiliaries $15.00. 120 pictures each. Free Catalog. PCS Software Box 18574, Kearns, UT 84118. (10/89)

130 XE, XEP80, I.C.D. MIO (256K) & CABLES. ALL DOCS. IN ORIGINAL CARTONS. USED LESS THAN 6 MONTHS. $375.00. ALSO ADAPTEC 4000A HD CONT. & 50 PIN SCSI CABLE, $60.00. AFTER 6 P.M. CDT 205-649-5004. (8/89)

Treasure Hunters Wanted to play a new action text game. Cash Prizes for info SASE MEADOWBROOK GAMES 30860 Alicante, Homeland, CA 92348 (8/89)

★★★★★★

10 disks with the best P.D. games ever made for the Atari 8-bit for only $10.00 plus 2.00 S/H. Order NOW! E.Lopez, P.O. Box 6772, Pico Rivera, CA 90661-6772

★★★★★★

The A Exchange—A Users Newsletter for ALL Atari Owners. Send SASE for FREE Issue. The A Exchange, P.O. Box 25852-A, Fresno, CA 93729. (8/89)

**EXCELLENT PD SOFTWARE FOR 800/XL/XE.** Large selection, low prices, and fast service. Send SASE: CLASSIC SOFTWARE, P.O. BOX 373, RUNNEMEDE, NEW JERSEY 08078 (8/89)

★★★★★★

**BEST PD DISKS/PRICES**—fastest service—FREE bonuses—8-bit/ST. Automatic DISK-A-MONTH program. Full disks less than $1.50 each. Super sample disk with catalog $3.50. Specify computer. PATNOR, POB 370782, El Paso, TX 79937

**ENJOY selected PD software!** Games—productivity—utilities—more! Demo disk, $3.00. . .catalog FREE. Homemade, 6011 Hyde Park Circle, Suite 208, Jacksonville, FL 32201 (8/89)

MUSIC LOVERS! MIDI YOUR ATARI! We have music software, Midi interfaces, and hundreds of pre-programmed songs available for the Atari XL/XE and Atari ST. For catalog send $1.00 to The MIDI Inn, P.O. Box 2362, Dept. AN7, Westmont, IL 60559. Specify your computer, music software and Midi instruments on hand. (9/89)

★★★★★★

**CREATE/PRINT MUSIC MANUSCRIPT.** 48K 800/XL/XE. Words and music. WYSIWYG. Epson, Star, Panasonic printers, or easily create your own driver. Very high quality display and output. A SERIOUS TOOL for musicians; a NEAT TOYfor music-lovers. Written by an M.A. in music. $19.95. 30-day MONEY-BACK GUARANTEE. John Kelleher, 4400 New Jersey Street, San Diego, CA 92116 (10/89)

Your choice of 8-bit P.D. software on each side of every disk. Great selection, competitive prices. Send SASE for catalog. Dino's RYO, 664 W.Arrow Hwy, Suite 197, San Dimas, Calif. 91773

CHRISTIAN SOFTWARE and PUBLIC DOMAIN PD low as $1.80/disk. Send SASE for Catalog. AGAPE GAMES Disk five Christian Games $12.95 (8 BITS W/48K). Dan Sharpe GPO Box 911 Chicopee MA 01021 (9/89)

USED 8-BIT EQUIPTMENT, SOFTWARE, ACCESSORIES, SEND S.A.S.E. (45 CENTS) BUSINESS SIZE, OR 45 CENT STAMPS FOR PRICES AND HUGE LIST. L. CHRISTOFFER, 1806 PATRICK, BURLINGTON, WA 98233 (8/89)

★★★★★★

Stop overpaying for P.D. Software!!! ALL 8-bit disks only $2.00 each, ST only $3.00. Quantity discounts, great selection. Specify computer. Send large SASE: MWPDS, 890 N. Huntington St., Medina, OH 44256 (8/89)

SI gives you MORE for LESS! Our SUPERIOR ATARI 800/XL/XE P-D Theme Collections average 35 programs (7 cents each!) per D-S disk. Shipping FREE. Send only $3.00 for MLX GAMES' GREATEST HITS (our Most Popular title)—receive SI's newest Catalog FREE! Catalog alone/ $2.00. SOFTWARE INFINITY, 642 East Waring Avenue, State College, PA 16801 (7/89)

**NEW 800/XL/XE SOFTWARE!** An International Collection of Shareware and Public Domain programs for the discriminating Atari owner. Every disk Guaranteed! Dependable world-wide service. Low prices! Free descriptive catalogue. BELLCOM, Box 1043-A, Peterborough, Ontario, Canada K9J 7A5 (8/89)

Atari 800/XL/XE public domain disks. High quality disks, fast service, free list. High quality disks, fast service, free list. 1st Byte, PO Box 130822, Tyler, TX 75713 (8/89)

250+ DS/SD DISKS OF PD SOFTWARE FOR YOUR ATARI 8-BIT. PLEASE SEND $3 TO: AAPDS 6-18TH EDGEMORE, HUTCHINSON, KS 67502.

Public Domain Software—ST $4/Disk, 800/XL/XE $3/Disk. Largest ST Selection. The Atari ST Book $16.95. Fast Service, Visa/Mastercard, FREE catalog—Specify Computer. BRE Software, 352 W. Bedford, Suite 104-A, Fresno, CA 93711. (800) 347-6760 (1/90)

★★★★★★

**BATTERY BACKED UP RAMDISK.** Never lose your memory, Boot from Ramdisk! No installation. Ramdisk cartridge plugs into Atari 400/800 or XL/XE. 32K $99.95, 64K $129.95, 192K $249.95 Certified check, Money order, VISA accepted. CANOE COMPUTER SERVICES, 11006-155 St., Edmonton, AB, T5P-2N3 Phone (403) 437-4619 (8/89)

# Advertisers Index



# Advertising Deadlines

## October 1989

Insertion Orders: June 30
Artwork: July 7
On Sale: Last week of August

## November 1989

Insertion Orders: August 1
Artwork: August 8
On Sale: Last week of September

# BACK ISSUE SALE

## Complete Your Atari "Toolkit"

## SAVE 25% TO 35%

### How To Order:

Simply give the Order Numbers of the issues you want, and include the letter corresponding to the following selections:

**A- magazine & disk**
**B- magazine only**
**C- disk only**

*Example: 1A- is for "April 1983, magazine & disk"*

If no letters are indicated in your order, we will automatically send you the magazine only (B).

### Prices:

| | 1 To 5 Issues | If you order more than 5 Issues |
|---|---|---|
| Magazine & Disk | $7.00 each | $6.50 each |
| Magazine only | $3.00 each | $2.75 each |
| Disk only | $5.00 each | $4.75 each |

Please include shipping & handling charges:
1 to 5 issues: $1 for each issue
6 and over : $ 5 plus 50c for each issue over 5.

**FOR ORDERS ONLY CALL TOLL- FREE**
**(800) 234-7001 (7-5pm PST.)**
**MasterCard & Visa Only**

Or send your check to: ANTIC BACK ISSUES
544 Second Street San Francisco, CA 94107
California residents add 6 1/2% sales tax.

## Back Issue Order Form

| Order # | Selection A, B, or C | Order # | Selection A, B, or C |
|---|---|---|---|
| ____ | ____ | ____ | ____ |
| ____ | ____ | ____ | ____ |
| ____ | ____ | ____ | ____ |
| ____ | ____ | ____ | ____ |
| ____ | ____ | ____ | ____ |
| ____ | ____ | ____ | ____ |
| ____ | ____ | ____ | ____ |

**Total Issues Ordered:** ____

| **Total Issues Ordered:** | **1 to 5** | **6 & over:** | |
|---|---|---|---|
| No. Issues of Sel. A: ____ | x $7 ea. | x $6.50 ea. | = $____ |
| No. Issues of Sel. B: ____ | x $3 ea. | x $2.75 ea. | = $____ |
| No. Issues of Sel. C: ____ | x $5 ea. | x $4.75 ea. | = $____ |

Total $ ____
Shipping & handling $ ____
Amount enclosed $____

❑ Check Enclosed ❑ VISA ❑ MasterCard

Name ____________________

Address ____________________

City __________ State ______ Zip ______

Credt Card # ____________________

Exp. Date____________________

Canada, Mexico and all other foreign add $1. per issue. Payment in U.S. funds must accompany all foreign orders. California residents add 6 1/2 % sales tax.

Antic
The ATARI Resource

## Missing any issues? Looking for those great utilities, games, and reviews? You'll find it all right here! Copies are limited, so order today!

| ORDER # | ISSUE |
|---|---|
| 1 | **April 1983:** Games—3-D Maze, Voyeur (No Disk) |
| 2 | **May 1983:** Telecomputing—Microids, Tele Chess (No Disk) |
| 3 | **June 1983:** Databases—Stargazing, Dancin' Man (No Disk) |
| 4 | **July 1983:** Adventure Games—Dragonsmoke, Shoot-em-up Math |
| 5 | **August 1983:** Graphics—3-D Fuji, Keystroke Artist |
| 6 | **September 1983:** Education—P/M Tutor |
| 7 | **October 1983:** Sports Games—AutoCassette |
| 8 | **November 1983:** Sound & Music—Air Raid, Casting Characters |
| 9 | **December 1983:** Buyer's Guide—AUTORUN.SYS, Automate Player/Missiles |
| 10 | **January 1984:** Printers—Pocket Calendars, Screen Dump |
| 11 | **February 1984:** Personal Finance—TYPO, Gauntlet |
| 12 | **March 1984:** International Issue—DiskRead, Poker Solitaire |
| 13 | **April 1984:** Games—Risky Rescue, Math Wizard |
| 14 | **May/June 1984:** Exploring XL Computers—Escape from Epsilon, Scroll to the Top |
| 15 | **July 1984:** Communications—AMODEM, BASIC Animation Secrets |
| 16 | **August 1984:** Disk Drives—Horseplay, Recall |
| 17 | **September 1984:** Computer Graphics—Graphics Converter, Olympic Dash (No Magazine) |
| 18 | **October 1984:** Computer Learning—Bouncing Ball, Antic 4/5 Editor/Animator |
| 19 | **November 1984:** Computer Adventures—Adventure Island, Advent X-5 |
| 20 | **December 1984:** Buyer's Guide—Infobits, Biffdrop |
| 21 | **January 1985:** Super Utilities—TYPO II, DISKIO |
| 22 | **February 1985:** Finances—Home Loan Analyzer, Drum/Bass Synth |
| 23 | **March 1985:** Printers—Kwik Dump, Font Maker |
| 24 | **April 1985:** Computer Frontiers—Dot Matrix Digitizer, Speech Editor |
| 25 | **May 1985:** New Super Ataris—Son of Infobits, Arena Racer |
| 26 | **June 1985:** Computer Arts—View 3-D, The Musician |
| 27 | **July 1985:** Computer Challenges—Miniature Golf, Guess That Song |
| 28 | **August 1985:** Telecommunications—Atari 'Toons, Pro*Term |
| 29 | **September 1985:** Power Programming—One-Pass Disk Copy 130, Crickets |

| ORDER # | ISSUE |
|---|---|
| 30 | **October 1985:** Mind Tools—Graph 3D, GEM Color Cascade |
| 31 | **November 1985:** New Communications—TYPO II Double Feature, 130XE Memory Management |
| 32 | **December 1985:** Shoppers Guide—DISKIO Plus, Box-In |
| 33 | **January 1986:** Atari Products are Back—Appointment Calendar, Dungeon Master's Apprentice |
| 34 | **February 1986:** Printer Power—T-Shirt Construction Set, Forth Escapes |
| 35 | **March 1986:** Practical Applications—Lunar Lander Constructor, Lie Detector |
| 36 | **April 1986:** Computer Mathematics—Fractal Zoom, 3-D Fractals |
| 37 | **May 1986:** 4th Anniversary—Digital Gardener, Molecular Weight Calculator |
| 38 | **June 1986:** Summer Computing—Weather Wizard, Bomb Squad |
| 39 | **July 1986:** Computer Arts—Amazing Card Shuffler, Grafcon ST |
| 40 | **August 1986:** Online Communications—Ultrafont, Floppy Filer |
| 41 | **September 1986:** Weather—WEFAX Decoder (8-bit/ST), BASIC Tracer |
| 42 | **October 1986:** Hard Disks—Video Stretch, TYPO ST |
| 43 | **November 1986:** Personal Finance—Budget dataBASE, V-Graph |
| 44 | **December 1986:** Shoppers Guide—Stepper Motors, Nuclear Waste Dump |
| 45 | **January 1987:** Talking Atari—Talking Typewriter, Rebound |
| 46 | **February1987:** Word Processing—SF Fogger, Electric Charlie! |
| 47 | **March 1987:** Dvorak Keyboard, Multi-AUTORUN |
| 48 | **April 1987:** —Designer Labels, Taxman |
| 49 | **May 1987:** 5th Anniversary—A-Rogue, Poker Slot Machine |
| 50 | **June 1987:** Animation—Verbot Commander, Citadel |
| 51 | **July 1987:** Print Anything—Ghost Writer, Your Net Worth |
| 52 | **August 1987:** Atari Muscle—Sideways Spreadsheet, Diamond Dave |
| 53 | **September 1987:** Work/Play—Mighty Mailer, Maximillian B. |
| 54 | **October 1987:** Football Predictor—Antic Prompter (8-bit/ST), Spelling Checker |
| 55 | **November 1987:** Practical Applications—Critical-Path Projects, WYSIWYG Cassettes |
| 56 | **December 1987:** Print Holiday—P.S. Envelope Maker, Antic Publisher |

# B&C ComputerVisions

3257 KIFER ROAD
SANTA CLARA, CA 95051
(408) 749-1003
(408) 749-9389 FAX

STORE HOURS
TUE - FRI 10am - 6pm
SAT - 10am - 5pm
CLOSED SUN - MON

## 800/XL/XE SOFTWARE
### ALL TITLES ON DISK

#### ENTERTAINMENT

| | |
|---|---|
| 12 ADAMS ADVENTURES | 14.95 |
| ALIANTS | 26.95 |
| ALT. REALITY CITY | 26.95 |
| ALT. REALITY DUNGEON | 26.95 |
| BEYOND CASTLE WOLF | 14.95 |
| BISMARK | 26.95 |
| BOP & WRESTLE | 26.95 |
| BORDINO:1812 | 26.95 |
| BOULDERDASH CONSTR.SET | 17.95 |
| BRUCE LEE | 17.95 |
| CASTLE WOLFENSTEIN | 14.95 |
| DALLAS QUEST | 7.95 |
| D-BUG | 7.95 |
| F-15 STRIKE EAGLE | 31.50 |
| FIGHT NIGHT | 17.95 |
| GAUNTLET (64K) | 31.50 |
| DEEPER DUNGEONS | 22.50 |
| GUNSLINGER | 26.95 |
| HARD HAT MAC | 7.95 |
| JAWBREAKER | 9.95 |
| KARATEKA | 13.50 |
| KNICKERBOCKERS | 13.50 |
| KORONIS RIFT | 13.50 |
| LAST V-8 | 8.95 |
| LEADERBOARD | 13.50 |
| MAIL ORDER MONSTERS | 13.50 |
| MICROLEAGUE BASEBALL | 35.95 |
| MONTEZUMA'S REVENGE | 14.95 |
| MOUSEQUEST | 17.95 |
| MOON SHUTTLE | 7.95 |
| NINJA | 8.95 |
| OIL'S WELL | 9.95 |
| O'RILEY'S MINE | 9.95 |
| PIRATES OF BARB. COAST | 22.50 |
| PREPPIE I & II | 9.95 |
| RESCUE ON FRACTALAS | 13.50 |
| SILENT SERVICE | 31.50 |
| SPEEDKING | 8.95 |
| SPIDERMAN | 7.95 |
| SPITFIRE 40 | 31.50 |
| STARFLEET I | 44.95 |
| SPY VS. SPY III | 17.95 |
| STOCKMARKET | 22.50 |
| STRIP POKER | 26.95 |
| SUMMER GAMES | 17.95 |
| TAX DODGE | 9.95 |
| THE HULK | 7.95 |
| TOMAHAWK (64K) | 26.95 |
| TOP GUNNER | 17.95 |
| TOUCHDOWN FOOTBALL | 13.50 |
| TRAILBLAZER | 26.95 |
| UNIVERSE | 44.95 |
| ZAXXON (400/800) | 13.50 |

#### PROGRAMMING

| | |
|---|---|
| ACTION! | 71.95 |
| ACTION! TOOLKIT | 26.95 |
| BASIC XL | 53.95 |
| BASIC XL TOOLKIT | 26.95 |
| BASIC XE | 71.95 |
| DOS 2.5 | 7.95 |
| DOS XE | 10.00 |
| DISK I/O | 26.95 |
| KYAN PASCAL | 62.95 |
| LIGHTSPEED C | 35.95 |
| LOGO | 19.95 |
| MAC/65 | 71.95 |
| MAC/65 TOOLKIT | 26.95 |
| MACRO ASSEMBLER | 22.50 |
| PILOT | 19.95 |
| SPARTA DOS X | 71.95 |
| TOP DOS 1.5 PLUS | 35.95 |

#### PRODUCTIVITY

| | |
|---|---|
| ANIMATION STATION | 89.95 |
| ATARIWRITER+ | 39.95 |
| ATARI BOOKKEEPER | 24.95 |
| ATARI MUSIC II | 14.95 |
| AWARDWARE (1050) | 13.50 |
| BANK STREET WRITER | 14.95 |
| BLAZING PADDLES | 31.50 |
| CELEBRITY COOKBOOK | 26.95 |
| COMPUTE YOUR ROOTS | 35.95 |
| DATAMANAGER | 17.95 |
| FAMILY FINANCE | 6.95 |
| GUITAR WIZARD | 26.95 |
| HOME ACCOUNTANT | 19.95 |
| HOME FILING MANAGER | 6.95 |
| HOMEPAK | 24.95 |
| INVENTORY MASTER | 80.95 |
| LETTER WIZARD | 29.95 |
| MUSIC CONSTRUCTION SET | 13.50 |
| NEWSROOM (1050 - 64K) | 44.95 |
| NEWS STATION | 26.95 |
| NEWS STA. COMPANION | 26.95 |
| PAGE DESIGNER | 26.95 |
| PRINT POWER (1050) | 13.50 |
| PRINTKIT (1050) | 13.50 |
| PRINTSHOP | 34.95 |
| P.S. COMPANION (64K) | 24.95 |
| P.S.GRAPHICS LIBRARY 1 | 17.95 |
| P.S.GRAPHICS LIBRARY 2 | 17.95 |
| P.S.GRAPHICS LIBRARY 3 | 17.95 |
| PROOF READER | 17.95 |
| PUBLISHING PRO | 35.95 |
| RUBBER STAMP | 26.95 |
| SYNTREND (130XE) | 35.95 |
| SUPER MAILER | 35.95 |
| THE LOTTO PROGRAM | 17.95 |
| TIMEWISE | 6.95 |
| TURBOWORD/80 COLUMN | |
| REQUIRES XEP80 | 44.95 |
| VIDEO TITLESHOP (64K) | 26.95 |
| GRAPHICS COMPANION | 17.95 |
| VIRTUOSO | 29.95 |
| VISICALC | 24.95 |

#### EDUCATION

| | |
|---|---|
| ATARI LIGHT MODULE | |
| (REQ. STARTER SET) | 9.95 |
| BUZZWORD | 35.95 |
| GRANDMA'S HOUSE (-10) | 9.95 |
| HEY DIDDLE (AGE 3-10) | 9.95 |
| MASTER TYPE | 14.95 |
| PLANATARIUM | 22.50 |
| STATES AND CAPITALS | 9.95 |
| TOUCH TYPING | 9.95 |
| **CBS** (AGE 3-6): | |
| ASTROGROVER | 8.95 |
| BIG BIRD SPEC DELIVE | 8.95 |
| ERNIE'S MAGIC SHAPE | 8.95 |
| **DESIGNWARE:** | |
| MATHMAZE (6-11) | 35.95 |
| MISSION ALGEBRA (13+) | 35.95 |
| SPELLICOPTER (6-11) | 35.95 |
| **TINK TONK** (AGE 4-6): | |
| ABC'S | 8.95 |
| COUNT AND ADD | 8.95 |
| SMART THINKER | 8.95 |
| SPELLING | 8.95 |
| SUBTRACTION | 8.95 |
| THINKING SKILLS | 8.95 |
| ALL 6 TINK TONKS | 39.95 |
| **UNICORN:** | |
| 10 LITTLE ROBOTS | |
| (PRE-SCHOOL) | 26.95 |
| FUN BUNCH (6-ADULT) | 26.95 |
| RACECAR RITHMETIC | |
| (AGE 6+) | 26.95 |
| **WEEKLY READER** (PRE-SCHOOL): | |
| STICKY BEAR SHAPES | 26.95 |
| STICKY BEAR NUMBERS | 26.95 |
| STICKY BEAR ABC'S | 26.96 |
| STICKY BEAR OPPOSITE | 26.95 |
| SB BASKET BOUNCE | 26.95 |
| STICKY BEAR BOP | 26.95 |
| RUN FOR IT | 26.95 |
| PIC BUILDER | 26.95 |

## 800/XL/XE SOFTWARE
### ALL TITLES ON CARTRIDGE

#### ENTERTAINMENT

| | |
|---|---|
| 3D TIC-TAC-TOE | 9.95 |
| AIRBALL (XL/XE) | 24.95 |
| ALIEN AMBUSH | 9.95 |
| ACE OF ACES (XL/XE) | 24.95 |
| ARCHON | 19.95 |
| ASTEROIDS | 15.95 |
| ASTRO CHASE | 14.95 |
| ATARI TENNIS | 9.95 |
| ATLANTIS | 14.95 |
| BALL BLAZER | 19.95 |
| BARNYARD BLASTER | |
| (REQ. LIGHT GUN) | 24.95 |
| BATTLEZONE | 19.95 |
| B.C. QUEST FOR TIRES | 19.95 |
| BLUE MAX | 19.95 |
| BOULDERS & BOMBS | 14.95 |
| CAVERNS OF MARS | 14.95 |
| CENTIPEDE | 14.95 |
| CHICKEN | 9.95 |
| CHOPLIFTER | 14.95 |
| CLAIM JUMPER (400/800) | 9.95 |
| CLOUDBURST | 9.95 |
| CRIME BUSTER | |
| (REQ. LIGHT GUN) | 24.95 |
| CROSSBOW | 24.95 |
| CROSSFIRE | 9.95 |
| CRYSTAL CASTLES (XL/XE) | 19.95 |
| DARK CHAMBERS (XL/XE) | 24.95 |
| DAVIDS MIDNIGHT MAGIC | 19.95 |
| DEFENDER | 14.95 |
| DELUXE INVADERS | 7.95 |
| DESERT FALCON | 19.95 |
| DIG DUG | 19.95 |
| DONKEY KONG | 5.00 |
| DONKEY KONG JR. | 19.95 |
| EASTERN FRONT (1941) | 19.95 |
| E.T. PHONE HOME | 9.95 |
| FIGHT NIGHT | 19.95 |
| FINAL LEGACY | 19.95 |
| FOOD FIGHT (XL/XE) | 19.95 |
| FOOTBALL | 14.95 |
| FROGGER | 14.95 |
| GALAXIAN | 9.95 |
| GATO | 24.95 |
| GORF (400/800) | 5.00 |
| GYRUSS | 14.95 |
| HARDBALL | 19.95 |
| INTO THE EAGLES NEST | 19.95 |
| JOURNEY TO PLANETS | 9.95 |
| JOUST | 19.95 |
| JUNGLE HUNT | 19.95 |
| KABOOM! | 14.95 |
| KARATEKA | 19.95 |
| KRAZY ANTICS | 14.95 |
| LODE RUNNER | 24.95 |
| MARIO BROS. | 19.95 |
| MEGAMANIA | 9.95 |
| MILLIPEDE | 14.95 |
| MISSILE COMMAND | 5.00 |
| MOON PATROL | 19.95 |
| MR. COOL | 9.95 |
| MS. PAC MAN | 19.95 |
| NECROMANCER | 19.95 |
| ONE ON ONE (XL/XE) | 19.95 |
| PAC MAN | 5.00 |
| PENGO | 19.95 |
| POLE POSITION | 19.95 |
| POPEYE | 14.95 |
| Q-BERT | 14.95 |
| QIX | 14.95 |
| RESCUE ON FRACTALAS | 19.95 |
| RETURN OF THE JEDI | 14.95 |
| ROBOTRON:2084 | 19.95 |
| SKY WRITER | 14.95 |
| SLIME (400/800) | 9.95 |
| SPACE INVADERS | 14.95 |
| STAR RAIDERS | 5.00 |
| STAR RAIDERS II | 19.95 |
| SUBMARINEE COMMANDER | 14.95 |
| SUMMER GAMES (XL/XE) | 24.95 |
| SUPER BREAKOUT | 9.95 |
| SUPER COBRA | 14.95 |
| THUNDERFOX | 19.95 |
| TRACK & FIELD | 24.95 |
| TURMOIL | 9.95 |
| WIZARD OF WOR | 5.00 |

#### PRODUCTIVITY

| | |
|---|---|
| ATARIWRITER | 19.95 |
| MICROFILER | 22.50 |

#### EDUCATION

| | |
|---|---|
| ATARILAB STARTER SET | 29.95 |
| FUN WITH ART | 14.95 |
| MATH ENCOUNTERS | 9.95 |
| **FISHER PRICE** (PRE SCHOOL): | |
| DANCE FANTASY | 8.95 |
| LINKING LOGIC | 8.95 |
| LOGIC LEVELS | 8.95 |
| MEMORY MANOR | 8.95 |
| **SPINNAKER** (AGE 3-10): | |
| ALF IN COLOR CAVES | 9.95 |
| ALPHABET ZOO | 9.95 |
| DELTA DRAWING | 9.95 |
| FACEMAKER | 9.95 |
| KIDS ON KEYS | 9.95 |
| KINDERCOMP | 9.95 |
| (AGE 7 - ADULT): | |
| ADV.CREATOR (400/800) | 9.95 |
| FRACTION FEVER | 9.95 |

## SUPER SPECIALS
### RECONDITIONED ATARI MERCHANDISE
### 30 DAY WARRANTY

| | | |
|---|---|---|
| **800 (48K) COMPUTER $79.95** | **SPACE AGE JOYSTICK $5.00.** | **1030 MODEM WITH EXPRESS! $24.95** |
| **400 (16K) COMPUTER $29.95** | **ATARI TRACKBALL $9.95** | **1010 PROGRAM RECORDER $29.95** |
| **1020 COLOR PRINTER/PLOTTER $19.95** 40 COLUMNS WIDE (new in box) | **ATARI BOOKKEEPER $14.95 - NO BOX** | **DISKETTES AS LOW AS 20 CENTS** 10 FOR $4.00 100 FOR $29.95 1000 FOR $200 MOST ARE UNNOTCHED WITH OLD SOFTWARE |
| | **ATARI NUMERIC KEYPAD $7.95** | |

**SHIPPING INFORMATION** - Prices do not include shipping and handling. Add $5.00 for small items ($8.00 Min. for Canada). Add $8.00 for disk drive. Add $2.75 for C.O.D. Calif. res. include 7% sales tax. Mastercard and Visa accepted if your telephone is listed in your local phone directory. Orders may be pre-paid with money order, cashier check, or personal check. Personal checks are held for three weeks before order is processed. C.O.D orders are shipped via UPS and must be paid with cash, cashier check or money order. International and APO orders must be pre-paid with cashier check or money order. $20.00 minimum on all orders. All sales are final - no refunds - prices are subject to change.

Phone orders accepted **TUESDAY THROUGH FRIDAY** from 10:00 am to 6:00 pm PST.

We carry a full line of **ATARI** products - large public domain library - write or call for **free** catalogue

**PRICES SUBJECT TO CHANGE WITHOUT NOTICE - ALL SALES ARE FINAL**